Defence Research and Development Canada / Recherche et développement pour la défense Canada

DEFENCE R&D DÉFENSE

Localization in self-healing autonomous sensor networks (SASNet)

Studies on cooperative localization of sensor nodes using distributed maps

Li Li

Defence R&D Canada – Ottawa

TECHNICAL REPORT

DRDC Ottawa TR 2008-020

January 2008

Canada

Localization in self-healing autonomous sensor networks (SASNet)

Studies on cooperative localization of sensor nodes using distributed maps

Li Li
Communications Research Centre

Defence R&D Canada – Ottawa

Technical Report
DRDC Ottawa TR 2008-020
January 2008

Principal Author

Original signed by Dr. Li Li

Dr. Li Li

Research Scientist

Approved by

Original signed by Joe Schlesak

Joe Schlesak

Manager, Defence Communications Program

Approved for release by

Original signed by Pierre Lavoie

Pierre Lavoie

Chief Scientist, Defence R&D Canada - Ottawa

The work described in this document was sponsored by the Department of National Defence under Work Unit 12pk

Abstract

The Self-healing Autonomous Sensing Network (SASNet) presents an advanced Wireless Sensor Network (WSN) that aims to enhance the effectiveness of mission operation in the contemporary military environment, by providing relevant and accurate situational awareness information. In order to achieve this objective, precise location information is required in SASNet. In this report, we present the studies conducted in the SASNet project on cooperative localization algorithms for wireless sensor nodes. We have taken the cooperative localization approach following the recommendations of the survey study conducted last year, which identified that cooperative localization schemes can often produce accurate results using a very small number of anchor nodes or even no anchor nodes.

The cooperative localization scheme adopted in this study computes a local map for each sensor node using all the available link metric constraints, and then merges the local maps into a global map where each node acquires its location coordinates. In the study, we examined the advanced techniques of non-linear data mapping for computing the local maps from the large data set of link constraints. In particular, we selected a non-linear mapping technique, the Curvilinear Component Analysis (CCA) from a class of highly efficient neural networks and applied it to WSN localization, proposing a novel cooperative localization scheme based on CCA. We studied CCA localization in comparison with another leading cooperative localization scheme, namely the MDS (Multi-Dimensional Scaling) map method.

In the report, we first briefly review the related work on WSN localization and re-examine the pros and cons of the selected cooperative approach vs. other approaches, most notably the iterative approach using trilateration. We then describe the CCA algorithm for data non-linear mapping, and extend it to solve the problem of sensor node position estimation. A detailed elaboration of the proposed CCA-MAP localization scheme is given. The performance simulations of CCA-MAP are conducted using SASNet scenarios and their results are illustrated and compared with the MDS-MAP algorithm, which is a leading cooperative localization scheme published in the literature. From the simulation experiments, advantages and shortcomings of the CCA-MAP algorithm are analyzed. Further, we discuss the design considerations of the discussed cooperative localization algorithms to compare and examine their implementation feasibility. Finally, conclusions and recommendations from this study are presented.

Résumé

Le réseau de capteurs autonomes à rétablissement automatique (SASNet) présente un réseau de capteurs sans fil qui vise à rendre plus efficace l'exécution des missions à l'aide d'information situationnelle utile et précise, dans le contexte contemporain des opérations militaires. Pour atteindre cet objectif, il faut introduire des données précises de localisation dans SASNet. Dans ce rapport, nous présentons les études menées dans le cadre du projet SASNet sur les algorithmes de localisation coopérative pour les nœuds capteurs sans fil. Nous avons adopté l'approche de localisation coopérative en réaction aux recommandations présentées à la suite de l'étude menée l'année dernière, qui stipule que les schémas de localisation coopérative peuvent souvent produire des résultats précis, avec un petit nombre de nœuds ancres ou sans nœud ancre.

Le schéma de localisation coopérative adopté dans le cadre de cette étude calcule une carte locale pour chaque nœud capteur à l'aide de toutes les contraintes de mesures de nœuds disponibles, pour ensuite fusionner les cartes locales dans une carte globale où chaque nœud capte ses coordonnées de localisation. Dans l'étude, nous avons examiné les techniques évoluées de mappage non linéaire de données pour calculer les cartes locales à partir du gros ensemble de données de contraintes de liens. Plus particulièrement, nous avons choisi une technique de mappage non linéaire, l'analyse de composants curviligne (Curvilinear Component Analysis (CCA)) dans une classe de réseaux neuronaux très efficaces, et nous l'avons appliqué à la localisation de réseau de capteurs sans fil, pour proposer un nouveau schéma de localisation coopérative fondé sur l'analyse de composants curviligne (Curvilinear Component Analysis (CCA)). Nous avons fait l'examen comparatif de la localisation d'analyse de composants curviligne (Curvilinear Component Analysis (CCA)), par rapport à un autre important schéma de localisation coopérative, notamment la méthode de mappage d'étalonnage multidimensionnel (Multidimensional Scaling (MDS)).

Dans le rapport, nous examinons d'abord brièvement les travaux connexes sur la localisation des réseaux de capteurs sans fil et examinons à nouveau les avantages et désavantages de l'approche coopérative sélectionnée en comparaison aux autres approches, surtout l'approche itérative à l'aide de la trilatération. Nous décrivons ensuite l'algorithme d'analyse de composants curviligne (Curvilinear Component Analysis (CCA)) pour le mappage non linéaire de données, et nous l'utilisons pour régler le problème d'estimation de la position des nœuds capteurs. On présente une description détaillée du schéma de localisation CCA-MAP proposé. Les simulations du rendement de CCA-MAP sont menées à l'aide de scénarios SASNet et les résultats sont illustrés et comparés à l'algorithme MDS-MAP, qui constitue un important schéma de localisation coopérative publié dans la documentation. À partir des expériences de simulation, nous avons pu analyser les avantages et inconvénients de l'algorithme CCA-MAP. De plus, nous abordons les facteurs à considérer au niveau de la conception des algorithmes de localisation coopérative pour comparer et examiner la faisabilité de la mise en oeuvre. Finalement, nous présentons les conclusions et recommandations de cette étude.

Executive summary

Localization in self-healing autonomous sensor networks (SASNet): Studies on cooperative localization of sensor nodes using distributed maps

Li Li; DRDC Ottawa TR 2008-020; Defence R&D Canada – Ottawa; January 2008.

Introduction or background: The Self-healing Autonomous Sensing Network (SASNet) presents a tiered Wireless Sensor Network (WSN). SASNet aims to enhance the effectiveness of individuals, small teams and sub-units in the contemporary military operational environment, by providing them with relevant and accurate situational awareness information. To achieve this, accurate location information is indispensable in the SASNet.

Two cooperative localization schemes are studied in this report, namely the MDS-MAP algorithm reported previously in the literature, and the new CCA-MAP algorithm that we have developed in the SASNet project. MDS (Multidimensional Scaling) and CCA (Curvilinear Component Analysis) are two different types of non-linear reduction methods. We have researched on the original CCA algorithm and devised an approach to apply it in the node position calculation. Extensive simulation studies were conducted on various SASNet scenarios employing MDS-MAP and CCA-MAP to compare and evaluate their performance. Then a preliminary design analysis is given to address the implementation requirements and issues, comparing CCA-MAP and MDS-MAP.

Results: In both MDS-MAP and CCA-MAP, each node computes a local map using MDS (in MDS-MAP) or CCA (in CCA-MAP) method. Then local maps are merged into a global map of the network in either *2D* or *3D* spaces. Both approaches produce fairly accurate position estimation results using none or a minimum number of anchor nodes. Comparing MDS-MAP and CCA-MAP, CCA-MAP generates more accurate position results in most of the SASNet deployment scenarios, especially when range measurements are available. Using ranging techniques to measure local distances between neighbouring nodes, CCA-MAP can contain the median error of the computed position to under *10%r* (*r* is the average radio radius) or even under *5%r*. Without ranging capabilities in the network, the accuracy of CCA-MAP degrades, though it can still produce position information with an acceptable average error ratio in most of the tested scenarios. For random irregular shaped networks, both CCA-MAP and MDS-MAP may fail to contain the median position error to under*10%r* for the desirable low node connectivity levels using only the connectivity information. Though this type of the random network where the sensor nodes are scattered aimlessly without any care of their supposed positions, is not a viable deployment scenario for SASNet, such a network example is presented in the report to illustrate their difficulties in obtaining accurate node positions.

The cooperative localization algorithm is often computational intensive. The cost of the CCA map computation is about $O(n^2)$ and MDS $O(n^3)$ given n nodes in the network map. Though the CCA data reduction method is rather efficient compared with other methods such as the MDS approach, the small size of the local map often diminishes the differences. In the studies where

our simulations were implemented in Matlab, we could not obtain sensible benchmarks of computing time expenses for the algorithms, as the loop execution of CCA-MAP has extremely poor performance in Matlab. We recommend porting of the implementation to a real programming platform using a practical programming language, e.g., C or C++, to evaluate the computational cost.

The implementation issues also include requirements for memory and messaging. Our preliminary analysis estimated the memory and messaging cost for the CCA-MAP and the MDS-MAP implementations. It has been found that the memory and messaging expense of CCA-MAP is comparable or less than that of MDS-MAP.

Significance: In this report, we present the studies on localization of networked sensor nodes. We have taken the cooperative localization approach following the recommendations from the survey study that was conducted last year. Cooperative localization algorithms often produce more accurate results using fewer number of anchor nodes, compared with other types of localization schemes. In fact, cooperative algorithms considered in this report are anchor free schemes that would be able to localize the nodes in a WSN without using any anchor nodes. Cooperative localization schemes may support both range-based and range-free networking situations, where the distance measurements between nodes are available in the former scenarios but not in the latter. These properties are quite desirable for military WSNs.

Future plans: In the future work, it would be required to identify the real computational cost of the CCA-MAP and MDS-MAP in the selected node platforms of SASNet, to determine which implementation option of the algorithm is viable. Experiments of the localization scheme on the selected node platforms need to be conducted for performance evaluation. A most suitable approach can then be selected considering the different resource capabilities of the SN and FN platforms.

Sommaire

Localization in self-healing autonomous sensor networks (SASNet): Studies on cooperative localization of sensor nodes using distributed maps

Li Li; DRDC Ottawa TR 2008-020; R & D pour la défense Canada – Ottawa; Janvier 2008.

Introduction ou contexte: Le réseau de capteurs autonomes à rétablissement automatique (SASNet) présente un réseau de capteurs sans fil à multiples niveaux. Le SASNet vise à rendre plus efficace les personnes, petites équipes et sous-unités dans le contexte contemporain des opérations militaires, en leur transmettant des données de localisation précises et pertinentes. Pour atteindre cet objectif, il faut introduire des données précises de localisation dans SASNet.

Ce rapport examine deux schémas de localisation coopérative, notamment l'algorithme MDS-MAP, mentionné auparavant dans la documentation, et le nouvel algorithme CCA-MAP que nous avons développé dans le cadre du projet SASNet. L'étalonnage multidimensionnel (Multidimensional Scaling (MDS)) et l'analyse de composants curviligne (Curvilinear Component Analysis (CCA)) constituent deux différents types de méthodes de réduction non-linéaire. Nous avons fait des recherches sur l'algorithme d'analyse de composants curviligne (CCA) original et conçu une approche qui permet de l'appliquer au calcul de la position des noeuds. On a mené des études de simulation approfondies sur divers scénarios de SASNet à l'aide de MDS-MAP et CCA-MAP pour comparer et évaluer leur rendement. On a ensuite présenté une analyse d'étude préliminaire pour satisfaire aux exigences et régler les problèmes de mise en œuvre, en comparant CCA-MAP à MDS-MAP.

Résultats: Dans le cas de MDS-MAP et CCA-MAP, nous appliquons des calculs de cartes locales et la fusion de cartes globales pour produire une carte du réseau, où chaque nœud fait l'acquisition de ses coordonnées dans des espaces *2D* ou *3D*. Les deux approches produisent des résultats d'estimation de la position relativement précis, à l'aide d'un nombre minimal de nœuds ancres ou sans nœud ancre. Dans le cadre d'une comparaison entre MDS-MAP et CCA-MAP, CCA-MAP produit des résultats de position plus précis dans la plupart des scénarios de déploiement du SASNet, surtout lorsque les mesures de distances sont disponibles. MDS-MAP présente des résultats acceptables dans certains cas. À l'aide de techniques de distance pour mesurer les distances locales entre les nœuds adjacents, CCA-MAP peut contenir l'erreur médiane de la position calculée à moins de *10%r* (*r* représente le rayon moyen de portée radio) ou même à moins de *5%r*. Si le réseau ne comporte pas de capacité d'évaluation de la distance, la précision de CCA-MAP se dégrade; il peut toutefois produire de l'information de localisation avec une marge d'erreur moyenne acceptable dans la plupart des scénarios mis à l'essai. Dans le cas des réseaux aléatoires de forme irrégulière, CCA-MAP et MDS-MAP ne comporteront possiblement pas d'erreur de la position médiane à moins de *10%r*, pour les niveaux de faible connectivité des nœuds souhaitables, si on se limite à l'information de connectivité. Malgré le fait que nous indiquons dans ce rapport que ce type de réseau aléatoire, où les nœuds capteurs sont éparpillés de manière aléatoire sans se soucier de leurs positions présumées, ne constitue pas un

scénario de déploiement viable pour le SASNet, on présente un tel exemple de réseau dans le rapport pour démontrer les difficultés à obtenir des positions de nœuds précises.

L'algorithme de localisation coopérative est plutôt vorace en calcul. Le coût de calcul de l'analyse de composants curviligne (CCA) de la carte est d'environ $O(n^2)$ et d'étalonnage multidimensionnel (MDS) est de $O(n^3)$, s'il y a n nœuds dans la carte. Même si la méthode de réduction des données de l'analyse de composants curviligne (CCA) est moins vorace au niveau des calculs que la méthode d'étalonnage multidimensionnel (MDS), la petite taille de la carte locale réduit souvent les différences. Dans le cadre des études sur la mise en œuvre de simulations dans Matlab, il était impossible d'obtenir des références suffisamment importantes au sujet de la durée de calculs des algorithmes, puisque l'exécution en boucle de CCA-MAP a présenté un rendement médiocre dans Matlab. Nous recommandons le transfert de la mise en œuvre sur une véritable plate-forme de programmation à l'aide d'un langage de programmation pratique, p. ex., C ou C++, pour comparer les coûts de calcul.

Les questions de mise en œuvre comprennent des besoins de mémoire, de messagerie et de calcul. La première option dégage les nœuds capteurs aux ressources limitées du calcul intensif de position, mais concentre le trafic de messages autour des nœuds de fusion. La deuxième option répartit le volume du trafic de messages; il impose toutefois la charge de calcul sur tous les nœuds capteurs, qui sont souvent très limités en matière de ressources.

Importance: Dans ce rapport, nous présentons les études menées dans le cadre du projet SASNet sur la localisation de nœuds capteurs en réseau. Nous avons adopté l'approche de localisation coopérative en réaction aux recommandations présentées à la suite de l'étude menée l'année dernière. Les algorithmes de localisation coopérative peuvent souvent produire des résultats plus précis, avec un nombre inférieur de nœuds ancres, comparativement aux autres types de schémas de localisation. En réalité, les algorithmes de coopération examinés dans ce rapport constituent des schémas sans ancre en mesure de localiser les nœuds d'un réseau de capteurs sans fil sans utiliser de nœud ancre. Les schémas de localisation coopérative peuvent soutenir les situations de réseautage fondées ou non sur la distance, où les mesures de distances entre les nœuds sont offertes dans le premier scénario, sans toutefois être offertes dans le deuxième scénario. Ces caractéristiques sont grandement souhaitables dans le cadre du réseau de capteurs militaires.

Perspectives: Dans le cadre de nos prochaines activités, il faudrait découvrir le coût de calcul réel de CCA-MAP et de MDS-MAP dans les plates-formes de nœuds sélectionnées de SASNet, pour déterminer laquelle des options de mise en œuvre d'algorithme est viable. Il faut faire des essais du schéma de localisation sur les plates-formes de nœuds sélectionnées, à des fins d'évaluation du rendement. On pourra ensuite choisir l'approche la plus appropriée, en tenant compte des différentes capacités de ressources des plates-formes de nœuds capteurs et de nœuds de fusion.

Table of contents

List of figures

List of tables

Acknowledgements

This work was undertaken in conjunction with the SASNet Technology Demonstration Project (12pk) which is supported by Defence Research and Development Canada. I would like to thank the SASNet team for the great opportunity working with them that supplied this entire process with interesting problems and excellent ideas. Particularly, I would like to thank SASNet project manager Louise Lamont for her guidance and support throughout this work, and SASNet team member Ying Ge for many great discussions and suggestions. The reviewers of this report, Prof. Thomas Kunz from Carleton University and Dr. John Robinson have provided many invaluable comments and advices that have greatly improved the technical contents and presentation quality of its final version. Their effort and time are truly appreciated.

1 Introduction

The Self-healing Autonomous Sensing Network (SASNet) is a Wireless Sensor Network (WSN) that aims to equip individuals, small teams and sub-units with relevant and accurate situational awareness information in the contemporary military operational environment. SASNet is envisioned to consist of a rapidly deployable large-scale sensor network where the complementary sensors are quickly emplaced such as by a person walking or riding in a vehicle. The sensor nodes after deployment will automatically discover each other and create the network in real time. The sensor nodes will then perform detection sharing of different types of sensing information to provide accurate situational awareness. Figure 1 depicts the architecture overview of SASNet.

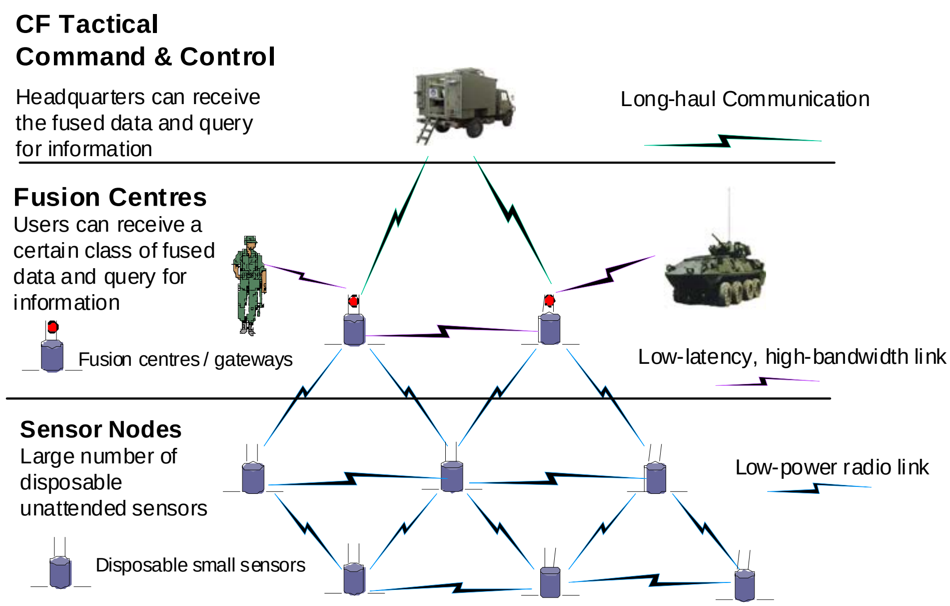

Figure 1: SASNet Architecture

SASNet is a hierarchical network with multiple layers of hybrid communication nodes as shown in Fig. 1. At level one, i.e., the sensor level, large numbers of unattended sensor nodes (SN) collect sensing measurements and transmit the information towards fusion nodes (FN) and gateways. At the fusion level (i.e., level two), the fusion nodes (FN) collect, aggregate and fuse the information for its immediate users that can be mobile agents or other nodes in the network. FNs also further send the required information to the higher-level command and control

headquarters. For details of the networking protocols and data dissemination mechanism in the hierarchical SASNet, please refer to [2] [3].

In the previous report of "Localization in Self-healing Autonomous Sensor Network (SASNet): Requirements, Design Considerations and Research Recommendations" [3], we have identified the requirements for node localization in SASNet that include to locate the static sensor nodes, and to locate and track the mobile agent node. This report studies the cooperative localization schemes to handle the first issue, i.e., to locate the static sensor nodes after their deployment. During the deployment of a SASNet system, some sensor nodes may be emplaced with known location coordinates obtained from either planning or GPS assistance. For example, a sensor node may be plugged into a GPS capable PDA to obtain its position coordinates on the spot right before its placement into its position. These nodes with known positions are the anchor nodes. Other rapidly deployed ad hoc sensor nodes that have no known coordinates will need to locate themselves in time to report readings associated with their location and to assist in tracking the mobile agents and targets.

The requirements elaborated in [3] have defined the problem of sensor node localization and its solution requirements to support the SASNet deployment process. Particularly a localization scheme that can achieve high accuracy containing the average node position error within *10%-20%* of the radius (r) of the radio signal range (i.e., *<10%r* to *20%r*) is of interest for scenarios of SASNet. The nature of the SASNet scenario demands a speedy deployment process. The cluttered and covered environment may cripple GPS while the fast deployment may not accommodate many nodes with planned locations. Thus, localization solutions that require many anchor nodes are not suitable [3]. Ranging measurements will be utilized to aim for high accurate distance readings applying spread spectrum or UWB technologies. However, distance measurements may not be relied on in all cases as complex terrain can impede the measurement accuracy. The localization solution of interest may assume thus the following characteristics: that it requires only a small number of anchor nodes to facilitate a rapid deployment process, and that it achieves a high level of position accuracy with or without the assistance of range measurements. Meanwhile, as pointed out in [3], because of the tiered architecture of SASNet, the resource constrains of the scanty sensor nodes have to be carefully considered. For example, messaging efficiency and computational efficiency of the localization algorithm are critical to preserve battery life of the small sensor nodes and to make the execution of the algorithm feasible. In [3], we selected the cooperative estimation algorithms to be the direction because of their better location estimate accuracy, and their often-small anchor node ratio requirements. In fact, the cooperative algorithms considered here are "anchor-free" localization schemes as they do not require any anchor nodes to derive the relative positions of the nodes, which are correct compared with the absolute position coordinates up to rotation, translation and reflection. In certain mission operations, relative position information can be sufficient. In other operations, if the location information needs to be used or compared with position information from other Cartesian coordinate system, only the minimum number (*3* for *2D* space) of anchor nodes would be required to perform the translation, rotation and reflection to attain the absolute position coordinates in the given Cartesian system.

Various localization schemes have been proposed in the literature as reviewed in [3]. Localization algorithms based on Multidimensional Scaling (MDS) [9][10][44][45][48] assume a class of close examples of the cooperative approach for deriving sensor node locations, which can be computed in a distributed or centralized manner in either range-based or range-free conditions with

minimum anchor nodes requirements, e.g., at least *3* (or *4*) anchor nodes in the *2D* (or *3D*) space. This class of algorithms also delivers higher node position accuracy *(<20%r*, *r* is the average radio signal radius) when compared with many other approaches. MDS is a non-linear mapping technique applying dimension reduction and data projection that transforms proximity information into a geometric embedding [50]. While preserving the distances between data points and reducing the data dimension from *N* to two (e.g., *2D* space) or three (e.g., *3D* space), MDS similar to other non-linear reduction algorithms, incurs fairly high computational cost of $O(n^3)$ and suffers quite often from local minima. As pointed out in [9][10][44][48], MDS is often good at finding the right general layout of the network, but not the precise locations of nodes. In [9][10][44][48], a refinement step using least-square minimization over the results obtained from MDS was applied to achieve better location accuracy. Compared with MDS mapping, the refinement process introduces an even higher computing cost, making it in an order of magnitude slower than the mere MDS algorithm. For example, more than *50* seconds may be taken to refine a map of *300* nodes on a *2GHz* Pentium powered laptop, compared to the MDS calculation in the first step that takes only less than a few seconds [10][44].

MDS based algorithms can achieve higher accuracy using much fewer anchor nodes than other approaches because it applies cooperative localization rather than trilateration. Cooperative localization jointly utilizes as much as possible the connectivity and distance information among all the nodes, while trilateration type of approach often employs only the distance information between nodes and the anchor nodes. Thus the latter usually requires higher volume of anchor nodes to ensure that every node can hear (and measure its distance to) at least three anchor nodes.

In studying cooperative localization approach such as MDS, we have compared various non-linear reduction techniques that might be applicable for node position estimation. An efficient neural network namely the Curvilinear Component Analysis [43]] was found that offers very accurate data dimension reduction while preserving the distances between the data points during the data reduction process, at a computational cost that is the least among the various reduction methods. Compared with most non-linear reduction algorithms that have a computational cost at $O(n^3)$, e.g., MDS [50] and Sammon's [52], CCA has its expense at $O(n^2)$. CCA is a self-organized neural network performing vector quantization and non-linear projection for dimensionality reduction and representation of multidimensional data sets [43]. Unlike general neural networks, CCA preserves the distances between the data points of the input data space in the output data space, and exhibits much higher efficiency in its unfolding of the dimensions. The non-linear projection capability of CCA turns out to be similar in its goal to other nonlinear mapping methods, such as MDS [50]] and Sammon's nonlinear mapping (NLM) [52], in that it minimizes a cost function based on inter-point distances in both input and output spaces. However, CCA in several aspects exhibits advantages that may be very useful in node localization process. Firstly, CCA overcomes fairly well the local minima problem to achieve improved accuracy. Localization algorithms using CCA can achieve desired accuracy without using any additional refinement process as that which was used in the MDS based localization algorithms. Secondly, CCA is more computational efficient and thus delivers a much faster data mapping process compared with other optimization processes [58][59]. Thirdly, being a neural network, CCA has the learning capability that can accommodate addition of new nodes [43][58] without re-running through the computing process for all the nodes. Other non-linear mapping algorithms such as MDS and Sammons would need to re-compute the entire data set.

We thus have selected the CCA data projection method and adapted it to the node localization process for wireless sensor networks. This report presents our studies and findings using the CCA algorithm to establish a cooperative localization scheme in SASNet. In the following, the CCA algorithm and our scheme to apply it in the sensor localization process are described in section 2. Section 2 also contains a brief overview of other localization schemes to provide the background information. In section 3, the performance studies of CCA and MDS for SASNet, and the comparisons between CCA and MDS based localization algorithms are presented. Practical design and implementation issues are discussed in section 4, followed by the conclusions and recommendations for future work in section 5.

2 Cooperative Localization Using CCA

Before elaborating on the non-linear mapping algorithm of CCA and its application in node location estimation, we provide a short review on related localization approaches and clarify in general the advantages and shortcomings of the cooperative approach that was selected. For a more detailed survey of the state-of-art of WSN localization, please refer to our previous report [3].

2.1 Related Work on WSN Localization

Various sensor node localization schemes have been proposed in the literature [3][4][5][7]-[23], [24]-[38], [39][40][42], [44]-[49], [53]-[57]. Based on whether using absolute measurements of node distances or alignment angles in the localization algorithms, the solutions can be classified as range-based [33][13][14][12][55] and range-free [32][7][9][26][27]. While range-based schemes use the absolute measurements in solving the location coordinates, range-free ones do not. It is generally true that range-based solutions often produce finer resolution results when the range errors are kept small. More accurate ranging technologies such as UWB [34] are found to offer promising measurement results. On the other hand, ranging techniques are very environment dependent, e.g., indoor vs. outdoor, path obstacles and complex terrains, etc., and often entail additional hardware cost. When measurements are not accurate, range errors may also have extended impact when accumulated during the position calculations. It has been reported that in certain cases, a range-free scheme may even outperform its range-based counterpart in estimated position accuracy due to the accumulated measurement errors incurred in the range-based approach [20]. In SASNet, though the latest ranging technologies would be explored to generate as much as possible the accurate measurement results, a solution that can be flexibly applied in both range-based and range-free scenarios is taken as a more versatile preference.

The second important aspect of any localization scheme is its algorithm. Position computation often applies trilateration, triangulation, or multilateration [12]. In a straightforward way, direct reach of at least three anchor nodes is needed for a node to compute its location coordinates [12][14][26]. When computing the position using any of the above methods, algorithms often employ iterations [12][53], starting from the anchor nodes in the network and propagating to all other free nodes to calculate their positions. One of the problems of this approach is its low success ratio when the network connectivity level is not high or when not enough well-separated anchor nodes exist in the network. To localize all the nodes, these algorithms quite often would require that *20%-40%* of the total nodes in the network be anchor nodes with average node connectivity level of about 10 or higher [7][27], unless anchor nodes can increase their signal range [26]. To solve the problem of demanding large numbers of anchor nodes, some approaches apply limited flooding to allow anchor nodes to be reached in multiple hops, and to use approximation of shortest distances over communication paths as the Euclidean distance [27]. However, such hop based distance approximation works rather poorly in anisotropic networks, introducing large position errors [7][27]. In many scenarios, they do not seem to significantly reduce the number of anchor nodes either [7][27]. High network connectivity levels required for the success of such algorithms also give rise to practical concerns, as dense neighbourhoods often severely impede radio network throughputs. In addition to the issues of large number of anchor nodes and high connectivity level, the accumulated location errors also need to be well dealt with

in this type of approach to maintain the accuracy of position estimates. Among such schemes, the one proposed in [53] reported one of the best results, where the position estimation error can be reduced to about *5%r* in more than *6* iterations when the network connectivity level is about *16* and *10%* of its nodes are anchors.

The cooperative localization schemes take a quite different approach, formulating the localization problem as joint estimation problems. Instead of using only the constraint between sensor nodes and anchor nodes, these solutions consider all constraints on inter-node distances and apply optimisation techniques to derive location coordinates. Algorithms based on rigidity theory are one kind of example [39] [56]. In [39], heuristics are employed to create a well-spread, fold-free graph layout that resembles the desired layout. Then a mass-spring model analogy is used to optimise the localization estimates using the minimum energy stage of the mass-spring model. Such an optimisation problem is NP-hard and would need further studies and proof on its convergence [39]. In [56], conditions for networks to be localizable using rigidity theory were investigated and a subclass of the grounded graphs were identified which can be computed efficiently. However, the focus was to find the network formation that can be computed. The overall performance of the algorithm for different network formations was not well reported. In [9][10][44][45][48] the node positions are calculated using the connectivity constraints applying multidimensional scaling (MDS). In [9][57], inter-node distance measurements are modelled as convex constraints, and then linear programming [9] and semi-definite programming (SDP) methods [57] were adopted to estimate the location of free nodes. These cooperative localization methods such as MDS and SDP are often quite powerful as they require a minimum number of anchor nodes and produce very accurate results. However, such algorithms are often computational intensive. In fact, it is doubtful if such algorithms can be executed on the mote sensor nodes [61], which is one type of the “disposable unattended sensor nodes” that might be applied in SASNet. Cooperative algorithms often tend to compute a map of a subnetwork or even the entire network rather than just the position of a single node during the process. Thus, the cooperative algorithm may be more suitable to be carried on the gateways or similarly powerful nodes rather than being executed on the small mote sensor node.

Comparing cooperative algorithms and iterative trilateration based algorithms, cooperative algorithms in general deliver more accurate position estimates and require fewer anchor nodes. Using cooperative algorithms, there is little overhead for computing the coordinates in *3D* space as compared to *2D* space, which is a nice property that iterative triangulation-based localization methods do not have. On the other hand, cooperative algorithms are relatively computational intensive. These algorithms handle better large data set, i.e., more intended for computing the locations of many nodes instead of for one node. It is more suitable thus for these algorithms to be executed on selected gateway nodes, rather than on the severely resource-constrained motes. The iterative trilateration based algorithm may be less computational intensive. Iterative trilateration / multilateration algorithms entail calculations of a much smaller linear system on each node through propagation of location calculations in the network. With good error control schemes, such algorithm as well can be accurate though it may require more anchor nodes, higher node connectivity levels and several rounds of location propagation and calculations to complete. Iterative algorithms naturally assume a fully distributed execution, intending to be executed on motes, though practically such real “mote implementations” are yet to be found. Simulations reported in the literature often take laptops to perform the iterative algorithms. For small mote sensors, the computation and messaging expenses (e.g., message propagations of several rounds in the network) of iterative trilateration algorithms can also be of concern.

2.2 Cooperative Localization Using Non-linear Mapping

2.2.1 Location Mapping Using MDS

Cooperative localization algorithms derive node positions in a generated map given certain constraints specified among the nodes, e.g., the distances between node pairs. There are often two possible outcomes when solving the localization problem using cooperative localization approach. One is a relative map and the other is an absolute map of the nodes. The relative map and the absolute map are only different in their coordinate systems. A relative map can be transformed to the absolute map using translation, rotation and reflection to conform to the coordinate system used by the absolute map. To transform a relative map to an absolute map, anchor nodes are required (e.g., at least *3* anchor nodes in *2D* space and *4* anchor nodes in *3D* space). Some applications would require an absolute map while others need only the relative map.

Non-linear mapping is one of the typical approaches used to resolve the non-linear constraints system. For instance, a distances matrix can be employed as a non-linear constraints system in a non-linear mapping solution to obtain the coordinates. One of the most popular cooperative localization schemes applies MDS non-linear mapping technique [9][10][44][45][48].

In MDS localization scheme, the network is represented as an undirected graph with vertices *V* and edges *E*. The vertices correspond to the nodes, of which zero or more may be special nodes, called anchors, with known positions. For the range free case, the edges in the graph correspond to the connectivity information. For the range-based case with known distances to neighbours, the edges are associated with values corresponding to the estimated distances. It is assumed that all the nodes being considered in the positioning problem form a connected graph. If an outlying node is not within communications range of any other nodes, it is not in the map nor will it have an estimated position.

The MDS localization algorithm is based on a well-established technique known as classical Multi-Dimensional Scaling (MDS). MDS has its origins in psychometrics and psychophysics. It is a set of data analysis techniques that display the structure of distance information data in a geometrical picture [50]. MDS starts with one or more matrices representing distances or similarities between objects and finds a placement of points in a low-dimensional space, usually two- or three-dimensional, where the distances between the points resemble the original similarities. MDS is often used as part of exploratory data analysis or information visualization. By visualizing objects as points in a low-dimensional space, the complexity in the original data matrix can often be reduced while preserving the essential information. Thus, when an N dimension distance matrix is scaled to two dimensions using MDS, the resulting two-dimensional data set represents the *2D* coordinates of the N nodes.

There are many types of MDS techniques, including metric MDS and nonmetric MDS, replicated MDS, weighted MDS, deterministic and probabilistic MDS. In classical metric MDS, proximities are treated as distances in a Euclidean space [51]. Analytical solutions are derived from the proximity matrix through singular value decomposition and provide the best low-rank approximation (e.g., *2D* space) in the least squared error sense. In practice, the technique tolerates error gracefully, due to the over determined nature of the solution. Because classical metric MDS

has a closed-form solution, it can be performed more efficiently than other forms of MDS on large matrices.

Classical MDS computes the coordinates *X* from a distance matrix *D* using singular value decomposition (SVD) on the double centered squared *D*. Double centering a matrix is subtracting the row and column means of the matrix from its elements, adding the grand mean, and multiplying it by *-1/2*. For a $n \times n$ *P* matrix of *n* points with *m* dimensions of each point, it can be shown that:

$$-\frac{1}{2}\left(p_{ij}^2 - \frac{1}{n}\sum_{i=1}^{n} p_{ij}^2 - \frac{1}{n}\sum_{i=1}^{n} p_{ij}^2 + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} p_{ij}^2 \right) = \sum_{k=1}^{m} x_{ik} x_{jk} \qquad (1)$$

Let's call the double centred matrix on the left-hand side of (1) *B*. Performing singular value decomposition on *B* gives us *B = UVU'* and coordinates $X = UV^{1/2}$. To compute the coordinates of nodes from *D*, The two largest singular values and singular vectors of *B* for *2D* networks and the three largest singular values and singular vectors of *B* for *3D* networks are taken. There is little overhead for computing the coordinates in *3D* space as compared to *2D* space, which is a useful property of the non-linear data reduction method. Note that matrix *B* is positive semi-definite, i.e., all its eigenvalues are real and non-negative, then the singular values and singular vectors coincide with the eigenvalues and eigenvectors.

We briefly describe the MDS-MAP algorithms here. For details of MDS localization, please refer to [9][10][44][45][48]. In this report, MDS-MAP refers to the distributed MDS localization algorithm, without including any centralized versions, as we do not consider the centralize algorithm. The steps of MDS-MAP are as follows:

1. For each node, neighbours within R_{lm} hops are involved in building its local map. The value of R_{lm} affects the amount of computation in building the local maps, as well as the quality. Often $R_{lm} = 2$ is used.

2. Compute local maps for individual nodes. For each node, do the following:

    a. Compute shortest paths between all pairs of nodes in its local mapping range R_{lm}. The shortest path distances are used to construct the distance matrix for MDS.

    b. Apply MDS to the distance matrix and retain the first *2* (or *3*) largest eigenvalues and eigenvectors to construct a *2D* (or *3D*) local map. The complexity of computing each local map is $O(k^3)$, where *k* is the average number of neighbours. Thus the complexity of computing *n* local maps is $O(k^3n)$, where n is the number of nodes. When computed in a distributed manner on each node in parallel, the computing complexity per node is $O(k^3)$.

    c. Refine the local map. Using the node coordinates in the MDS solution as the initial point, least squares minimization is performed to make the distances between nearby nodes match the measured ones. The computational expense of refinement is $O(k^3n)$. In the prototype implementation using Matlab, this refinement step is more

computationally expensive than MDS due to the poor performance of “loop” execution in Matlab.

3. Merge local maps. Any node can be selected to perform the merge, though practically, certain nodes in the network that have more computing power or need to know positions of other nodes can be selected to construct the global map. Use a randomly selected starting node’s local map as the starting current map. Each time, the neighbour node whose local map shares the most nodes with the current map is selected to merge its local map into the current map. Two maps are merged using the coordinates of their common nodes. The best linear transformation (minimizing discrepancy errors) is computed to transform the coordinates of the common nodes in one map to those in the other map. To merge a new local map *B* into the current map *A*, a linear transformation (translation, reflection, orthogonal rotation, and scaling) is determined to ensure that the coordinates of the common nodes in map *B* after transformation are best conformed with those in current map *A*. That is, given the coordinates of common nodes in maps *A* and *B* as matrices X_A and X_B, the linear transformation $T(.)$ delivers minimum sum of squared errors, i.e., $\min_T \|T(X_B) - X_A\|_2$ to merge map *B* into *A*. In section 4, we will experiment the map merge by using different starting nodes and different map merge orders to explore the impact of the merge order.

   The complexity of this step is $O(k^3n)$, where k is the average number of neighbours and n is the number of nodes.

4. Refine the global map (optional). Using the node coordinates in the global map as the initial solution, least squares minimization is applied to make the distances between neighbouring nodes match the measured ones.

   This step costs $O(n^3)$ and is much more expensive than the other steps for large networks.

5. Given sufficient anchor nodes (*3* or more for *2D* networks, *4* or more for *3D* networks), transform the global map to an absolute map based on the absolute positions of anchors. For r anchors, the complexity of this step is $O(r^3 + n)$.

2.2.2 Location Mapping Using CCA

Before elaborating on our localization scheme applying Curvilinear Component Analysis (CCA), we first briefly discuss the non-linear data mapping method of CCA [43].

2.2.2.1 Non-linear Projection Using CCA

Given N input vectors $\{x_i; i = 1,..., N\}$ where each vector x_i is of n dimensions, CCA looks for N output vectors $\{y_i; i = 1,..., N\}$, where each y_i is of s dimensions ($s < n$). Additionally, the distance between input vectors x_i and x_j is preserved between output vectors y_i and y_j. That is, given the Euclidean distance between x_i's as: $X_{ij} = d(x_i, x_j)$ and the corresponding distance in the output space $Y_{ij} = d(y_i, y_j)$, CCA pushes Y_{ij} to match X_{ij} for every pair (i, j) while minimizing a cost function

$$E = \frac{1}{2}\sum_{i}\sum_{j \neq i}(X_{ij} - Y_{ij})^2 F(Y_{ij}, \lambda_y) \tag{2}$$

Here X_{ij} forms a $N \times N$ distance matrix of $X_{ij} = d(x_i, x_j) = \sqrt{\sum_{k=1...n}(x_{ik} - x_{jk})^2}$ if Euclidean distance is taken, though CCA does not limit it to the Euclidean distance but allows any defined distance values; the same is true for Y_{ij}. $F(Y_{ij}, \lambda_y)$ is a weighing function, often bounded and monotonically decreasing in time (i.e., decreasing with each computing cycle). Decreasing exponential, sigmoid, or Lorentz functions are all suitable choices for *F*. We choose *F* as an exponential function for our experiments as shown in the next subsection.

In its minimization process of the cost function (2), CCA applies a different approach than most other methods (e.g., gradient descent methods) to improve the computing efficiency as described in [43]. Its update rule for each cycle is much simpler by pining one y_i and moving all the other y_j around. Only the distances from the node *i* to the other *N-1* nodes need to be computed, instead of all the $N(N-1)/2$ distances in both the input and output spaces. This is different from other methods such as the stochastic gradient descent ($\Delta y_i \approx -\nabla_i E$) or the steepest gradient descent where one vector y_i is moved every time according to the aggregated influence of every other y_j. The update for each cycle in CCA is thus the following simple step:

$$\Delta y_j = \alpha(t)F(Y_{ij}, \lambda_y)(X_{ij} - Y_{ij})\frac{y_j - y_i}{Y_{ij}} \quad \forall j \neq i \tag{3}$$

Here $\alpha(t)$ decreases with time similarly to usual stochastic gradient methods. For an adaptation cycle of all nodes (except *i*), the complexity is therefore only *O(N)* instead of $O(N^2)$ as in most other NLM algorithms. This not only converges the computation much faster, but also makes it more likely to eventually escape from local minima to reach a much deeper minimum as illustrated in [43].

2.2.2.3 Node Coordinates Projection

We formulize the localization problem as the following: Given a distance matrix $D_{(N \times N)}$ of *N* nodes, find the coordinates of all the nodes to achieve:

$$\min \sum_{i,j}(d_{ij} - p_{ij})^2 \quad \text{for } i,j=1,2,\ldots N \tag{4}$$

where

d_{ij} is the measured/known distance between node *i* and *j*, and

p_{ij} is the distance between node *i* and *j* computed using the calculated position coordinates of *i* and *j*.

If d_{ij} is taken as the distance matrix of the input data set and p_{ij} the distance matrix of the output data set, CCA then pushes (4) to a minimum as it minimizes the cost function in (2).

As the only known data of these *N* nodes assumed here is their distance matrix $D_{(N\times N)} = (d_{ij})_{(N\times N)}$, we take the distance matrix $D_{(N\times N)}$ as both the input data set, i.e., $x_{(N\times N)} = D_{(N\times N)}$ and the inter-vector distance matrix of the input data set, i.e., $X_{ij} = D_{(N\times N)}$. Even though $D_{(N\times N)}$ is not the real distance between vectors (i.e., the row vectors) in $D_{(N\times N)}$, the CCA algorithm, being a neural network, projects data points quite well given a defined distance matrix without requiring that it be the real Euclidean distance between the input data vectors.

In this way, we have now in the input space *N* vectors $\{x_i; i = 1,..., N\}$ each of *N* dimensions (i.e., *n*=*N*). The output data set contains *N* vectors each reduced to a dimension of 2 (or 3). Without losing the generality of applicability to both *2D* and *3D* spaces, we use a dimension of 2 in the following discussions. Thus the output data set is denoted as $y_{(N\times 2)}$ which is in fact the *2D* coordinates matrix of the *N* nodes. It is worth noting that unlike other methods such as iterative trilateration, cooperative algorithms using NLM including MDS, CCA, etc. often incur very little extra cost computing positions in *3D* space compared with that in *2D*. This property makes the NLM based cooperative algorithm advantageous for real network deployments.

Our CCA algorithm consists of the following two simple steps to project node coordinates given their distance matrix:

1. Set the initial output estimation of $y_{(N\times 2)}$ using the mean values of the first two columns of the input data set $x_{(N\times N)}$, adjusted by a uniformly randomized standard deviation of the same column.

2. In each cycle, select node *i* and compute for each node j $(j \neq i)$ the new $y_j(t+1)$ from the current value of $y_j(t)$ using:

$$y_j(t+1) = y_j(t) + \alpha(t)e^{-\frac{Y_{ij}}{\lambda(t)}}(\frac{X_{ij}}{Y_{ij}} - 1)(y_j - y_i) \tag{5}$$

Here we selected $F(Y_{ij}, \lambda) = e^{-\frac{Y_{ij}}{\lambda(t)}}$. Both $\alpha(t)$ and $\lambda(t)$ decrease with time (i.e., along each computing cycle). In our experiments, the following function (equation (6)) is used to implement

(i.e., using $\alpha = \nu$ and $\lambda = \nu$ in (6) below) $\alpha(t)$ and $\lambda(t)$, though other similar functions can be selected instead:

$$\nu(t) = \nu(0) \times \left(\frac{\nu(c)}{\nu(0)} \right)^{\frac{t}{c-1}} \tag{6}$$

where c is the number of total computing cycles, which is often also called the training length in CCA. We choose the following values in our experiments:

$\alpha(0) = 0.5$, $\alpha(c) = \alpha(0)/100$, $\lambda(0) = \max\{std_1, std_2, ..., std_N\} \times 3$ and $\lambda(c) = 0.01$, where std_i is the standard deviation for the i^{th} column of the input data set $D_{(N \times N)}$.

The number of training cycles required is mainly related to the size of the input data set and also the "goodness", i.e., the accuracy of the distance matrix. The bigger the input data set, i.e., the larger the *N*, the fewer the cycles are required projecting the final output data. A more accurate distance matrix would also result in fewer cycles, as the cost function of (1) will decrease much faster towards the minimum. A maximum number (e.g., 100 is used in our experiments for computing local maps) may be assigned to the total cycles allowed in the algorithm. During the execution, if we have the cost function $E < \varepsilon$ (e.g., ε is averaged to about 10^{-5} in range-based cases in our experiments) before reaching the maximum number of cycles, the algorithm exits and the projected data form the final output data set.

Though the above description applies CCA to the task of determining node locations, the distance matrix used as the input for the algorithm is often not available for large networks. Recently, Drineas et al. [47] have proposed algorithms for distance matrix reconstruction for sensor network localization using single value decomposition. This may provide an option to obtain the distance matrix for the network. We however assume that the distance matrix of the network is unknown. Instead, a distributed map algorithm [44] is adapted in the scheme to compute the node coordinates in the network.

2.2.2.4 Distributed Map Algorithm Using CCA

Adopting steps similar to the distributed map algorithm of MDS-MAP [10], we propose an alternative distributed map algorithm, the CCA-MAP algorithm. For simplicity, in the rest of the paper, we use MDS-MAP to refer to the distributed MDS localization scheme, i.e., MDS-MAP(P) and MDS-MAP(P,R) [10][44] because only the distributed version of MDS-MAP is considered in this report.

The CCA-MAP scheme similar to MDS-MAP builds local maps for each node in the network and then merges them together to form a global map. Differently from MDS-MAP, CCA is employed in computing the node coordinates in the local map.

Each node computes its local map using only the local information. If accurate ranging capability is available in the network, local distance between each pair of neighbouring nodes is measured

and known. Otherwise, only connectivity information is applied to assign value *1* to the edge between each neighbouring pair of nodes. Then a distance matrix for all the nodes in the R_{lm} hop neighbourhood of node x can be constructed using the shortest distance matrix as the approximation. Instead of a fixed $R_{lm}=2$, as that used in MDS-MAP algorithms [10][44], R_{lm} may be set adjustable in CCA-MAP. CCA reduction technique can generate quite accurate results with a reasonably accurate distance matrix of a small size, for example, a distance matrix of 12×12 or bigger. Therefore, in the ranging based scenarios where the local distance measurements are known with a certain level of accuracy (e.g., *5%* local distance measurement error), the one hop neighbourhood distance matrix of a certain size, e.g., size $>12 \times 12$, not only is more accurate than the two hop distance matrix of approximation, but can also be computed faster using CCA to produce more accurate position estimates. In the range-free options, a bigger size of the distance matrix assists better in mapping to the node position coordinates using CCA. Thus in the range-based option of the algorithm, for any given node, if its one hop neighbourhood has more than 12 nodes, R_{lm} =1 may be chosen for improved performance. Otherwise, R_{lm} =2 is applied. In the range-free computations of the local map where the local distance matrix is particularly inaccurate using the hop count approximation, often R_{lm} =1 is only taken when the one hop neighbourhood expands to a much larger size of *30* or *40*.

The steps of CCA-MAP can be described as follows:

1. For each node, neighbours within R_{lm} hops are included in building its local map. Compute the shortest distance matrix of the local map and take it as the approximate distance matrix *LD*.

2. Each node applies the CCA algorithm, using the local distance matrix *LD* as both the input data set and the distance matrix of the input data set as described in the previous subsection. This generates the relative coordinates for each node in the local map of node x of its R_{lm} hop neighbourhood. The complexity of computing each local map is $O(k^2)$, where k is the average number of neighbours. Thus the complexity of computing n local maps is $O(k^2 n)$, where n is the total number of nodes in the network. In comparison, the cost of computing n local maps in MDS-MAP is $O(k^3 n)$. MDS-MAP then may apply a local refinement process on each local map, which has an additional computational cost of $O(k^3 n)$. If local maps are computed in the distributed manner on each of the n nodes, the time complexity of local map computing is $O(k^2)$.

3. Merge local maps as described in the previous subsection for the MDS-MAP algorithm. The complexity of this step is $O(k^3 n)$, where k is the average number of neighbours and n is the number of nodes. The cost here is same as that of MDS-MAP. After map merge, MDS-MAP may apply a global refinement process whose computational cost is $O(n^3)$, which tends to dominate overall computational complexity for large networks. Note that the refinement processes of MDS-MAP for both the local maps in step 2 and the global map here are optional. We also have found that these extra refinement processes may not necessarily improve the node position estimate accuracy for many scenarios.

4. Given sufficient anchor nodes (*3* or more for *2D* space and *4* or more for *3D* space), transform the merged map to an absolute map based on the absolute positions of the anchor nodes. For r anchors, the complexity of this step is $O(r^3 + n)$.

In the CCA-MAP scheme, no refinement is applied because the results are often satisfactory without further optimization.

The computing of the local map can be distributed at each local node, or can be carried out at more powerful gateway nodes of each cluster should the sensor network have a hierarchical structure to relieve the severely resource-limited sensor nodes from any of the computing and communication demands imposed by localization. The local maps can be merged in parallel in different parts of the network by selected nodes. There is no need for anchor nodes in merging the maps. When at least three anchor nodes are found in the merged map of a subnetwork, an absolute map of the subnetwork can be computed using the coordinates of the anchor nodes to obtain the absolute coordinate values of all the nodes in the map of the subnetwork. The options for the distribution of the algorithm computation and their related messaging requirements are discussed later in Section 4.

3 Localization Experiments for SASNet Scenarios

In this section, we apply MDS-MAP and CCA-MAP to SASNet deployment scenarios to evaluate the performance of the two localization schemes.

3.1 Scenarios and Simulation Environments

The key SASNet scenarios are defined in the SASNet requirement document [1] and the SASNet demonstration plan [2]. In the scenarios, ASNet is deployed to monitor perimeters, road conjunctions, and choke points and to provide early warning of enemy approaches. The deployment topology for different scenarios and other configuration parameters (e.g., network sizes) are discussed in the above two documents [1][2].

In the scenarios for perimeter monitoring, full loop and partial loop protection deployment were specified in [1] with partial loop as the preferred option. The size of the network was specified at about *60* sensor nodes or more [1]. Too many sensor nodes are not favoured, as that would hinder the speedy deployment. For the perimeter-monitoring scenario, the C-shape network is selected for the experiments.

For road conjunction monitoring, up to five different sections/cluster-groups may be deployed. Each of the sections assumes a network that may be modelled as either a rectangle or a square. The size of each section was specified at about *50-100* sensor nodes. Though the sections may be interconnected for communications, a map is required for each of the sections. Localization is thus performed separately for each section. This implies that each section has at least three anchor nodes of its own in order to generate an absolute map for the section. Different sections will not share the anchor nodes because anchor nodes over long distance will not deliver accurate distance measurements or position estimates.

To monitor a choke point, a network of about *50* or more sensor nodes may be required as specified in [1]. The topology/shape of the deployed area can be modelled as a rectangle or a square or irregular. The scenarios of early warning provisioning often contains multiple network sections /cluster-groups. Each of these cluster groups is in fact quite similar to the other scenarios as described above.

In summary, the following network configurations are selected in the experiments: the C-shape network as a partial loop of perimeter monitoring; the rectangle-shaped network for roadside and approach area monitoring; and the irregular (close to square) network for road conjunction and choke point monitoring.

In summary, the following network configurations are selected in the experiments: the C-shape network as a partial loop of perimeter monitoring; the rectangle-shaped network for roadside and approach area monitoring; and the irregular (close to square) network for road conjunction and choke point monitoring.

In addition to the different sizes of the sensor networks and the different topologies / shapes of the network coverage, the sensor nodes may be randomly scattered, or more carefully planted during the deployment, corresponding to the random network and the grid network. In our experiments, it has been found that a random network often requires relatively higher average node connectivity levels to get connected. When connected, the node connectivity levels are often very unbalanced; with some nodes having a large number of neighbours and others a couple of connected links. Such a network is very vulnerable to failure and prone to network partitions if even a single link failure is incurred. Figure 2 below illustrates the connectivity of two network examples that cover the same area of 10×10. Network (a) has nodes randomly scattered and (b) has nodes placed approximately along grids with about *10%* node placement error. The random network (a) contains twice as many nodes as that in network (b) and has much unbalanced network connectivity. The random network in Figure 2 (a) only gets connected when the average node connectivity level reaches *CL=8.39*. It can be see in the Figure 2 (a) that while some nodes have only one or two links, other nodes have quite heavily populated neighbourhood. The network in Figure 2 (b) on the other hand gets fully connected at an average node connectivity level of *CL=2.88*. At *CL=5.98* as shown in Figure 2 (b), the connected radio links are quite evenly distributed among all the nodes, creating no area that may be threatened by single point of failure. The performance of radio networks degrades when the node connectivity level is too high because of the crowded neighbourhood. The network is also very vulnerable when the node connectivity level is too low. Generally, the node connectivity at about *CL=6* is preferred for better radio communications [18]. A network with very unbalanced node connectivity levels is not favoured. Therefore, though many claim and fancy the random scattering of sensor nodes in a WSN, a more practical approach would still require that the sensor node be placed with some care close to the desirable and required locations.

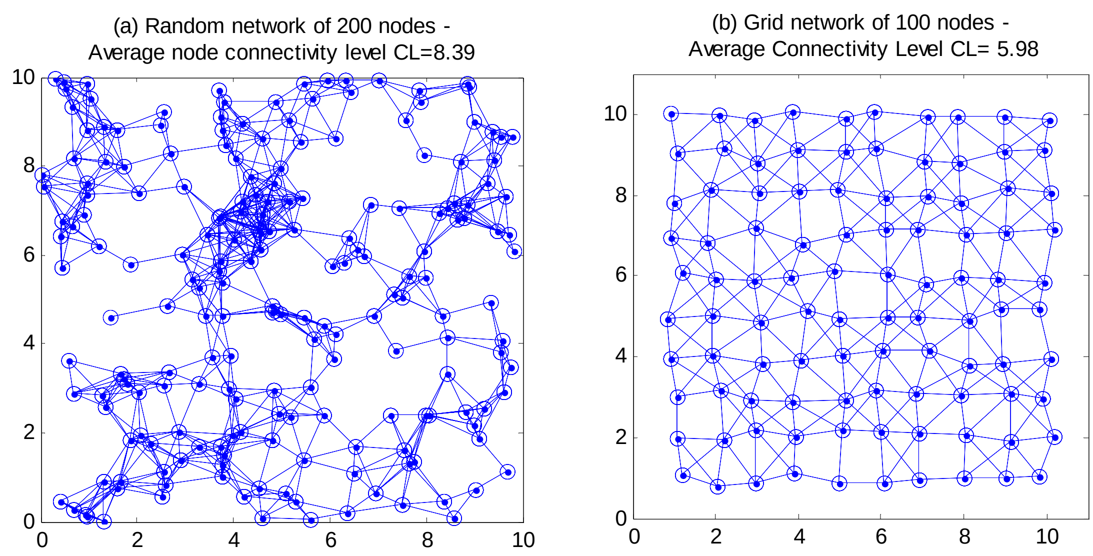

Figure 2. Network connectivity comparison of random and grid networks

However, the well-formed grid network often requires a somewhat lengthy deployment process depending how precisely the grid should be formed. To achieve a fairly fast deployment process

that can still establish a reasonably well-formed network, near grid placement of node with tolerable error on the placement positions may be one of the options. Thus in the experiments presented in this report, the grid like network is often selected. In these networks, each node is deployed to a grid point with the placement error uniformly distributed in the range of [*-20%i, 20%i*], where *i* is the unit length. The random network cases are also thoroughly examined in our studies [4], though only a couple of random network examples will be covered here for the purpose of discussions of performance evaluations.

We currently do not have yet the final decision on the number of sensor nodes that would be under the governance of one fusion node. In the map generation process using either MDS or CCA, there is no difference in the algorithm when computing the position of a sensor node or a fusion node. Where the algorithm should be implemented and executed on the other hand would be determined by the different capabilities of the different types of nodes. These implementation issues will be discussed in the next section.

During the experiments, for each network type, the algorithms were run on multiple randomly generated network samples. Several sets of differently positioned anchor nodes are selected for each network sample to examine the effects of anchor nodes. Both range-based and range free options are tested. When using the range free option, hop count is used as the distance matrix between a pair of nodes as described before. In the range-based scenarios, the range is modelled as the actual distance combined with Gaussian noise. Thus, the range measured between neighbouring nodes is a random value drawn from a normal distribution with actual distance as the mean and a standard deviation of *5%*.

In the following discussions, MDS-MAP refers to the distributed MDS-MAP algorithm as indicated in the previous section. Though originally MDS-MAP includes the centralized version, we do not consider the centralized map algorithm in this project.

3.2 C-shape Networks

3.2.1 Network Experiment Parameters & Configurations

A C-shape grid network of 79 nodes with node placement error uniformly distributed in the range of [*-20%i, 20%i*] is employed in this experiment. The network area consists of a $10r \times 10r$ square with $i = 1$ set as the placement unit length. Selecting anchor nodes at various positions in the network, both CCA-MAP and MDS-MAP schemes have been tested. The algorithms were run on multiple randomly generated network samples

Figure 3 depicts the anchor node positions when *3* anchor nodes are deployed with known location coordinates in the network. Figure 4 and Figure 5 illustrate the different sets of anchor positions selected when *4* and *6* anchor nodes are deployed in the network. Experiments were conducted across C-shape network scenarios that consist of these different anchor sets of different numbers of anchor nodes. Multiple network samples of the C-shape network were generated. For each of the network samples, the algorithms were run using each of the anchor sets as shown here in the Figures.

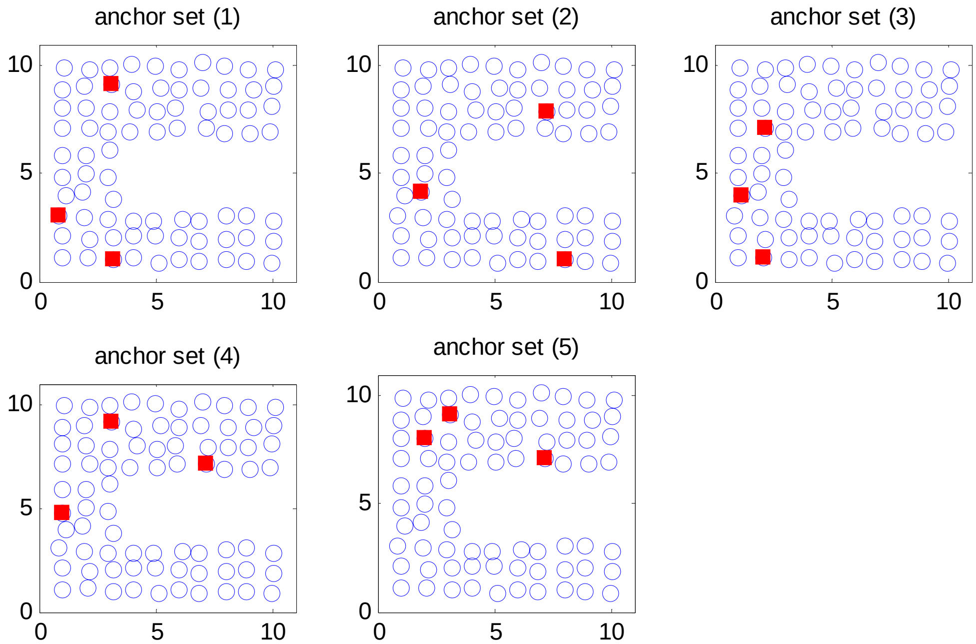

Figure 3 C-shape network scenarios with three anchor nodes

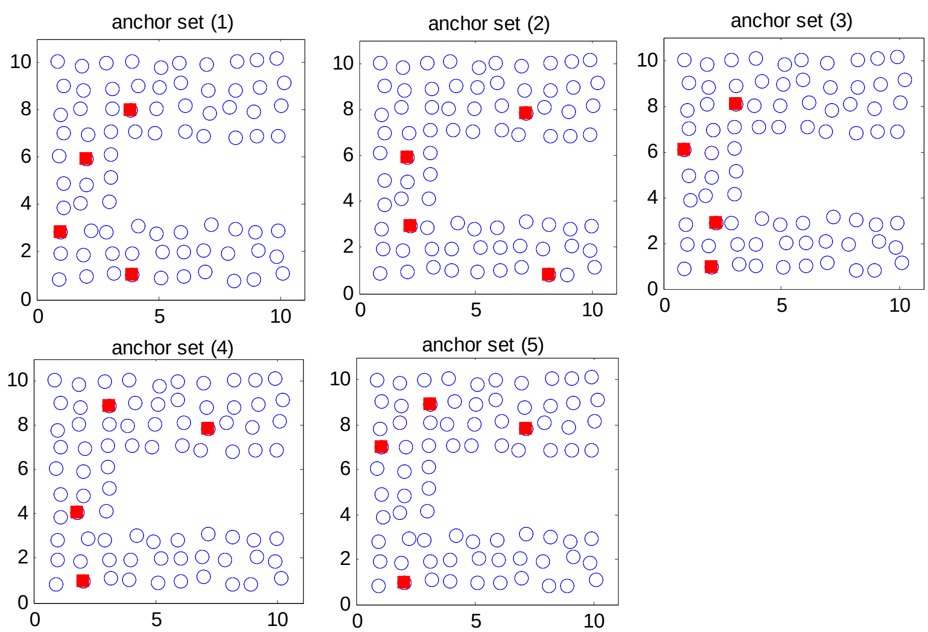

Figure 4 C-shape network scenarios with 4 anchor nodes

In the algorithms of both CCA-MAP and MDS-MAP, local maps are computed first for each node. The local maps are then merged into a global map. Next the merged global map is translated/rotated into the absolute global map using the known coordinates of the anchor nodes. Each node acquires its coordinates in the global map. When merging the global map, the local map selected as the starting local map can be different. Often, the starting local map is randomly chosen. In the experiments, different local maps are tested as the starting merging map. Note that these are the nodes whose local maps are used as the starting map for merging. They are not necessarily the nodes that start executing the map merging. In fact, map merging can be carried on any node selected based on the computing power, communication bandwidth and other resource capabilities of the node.

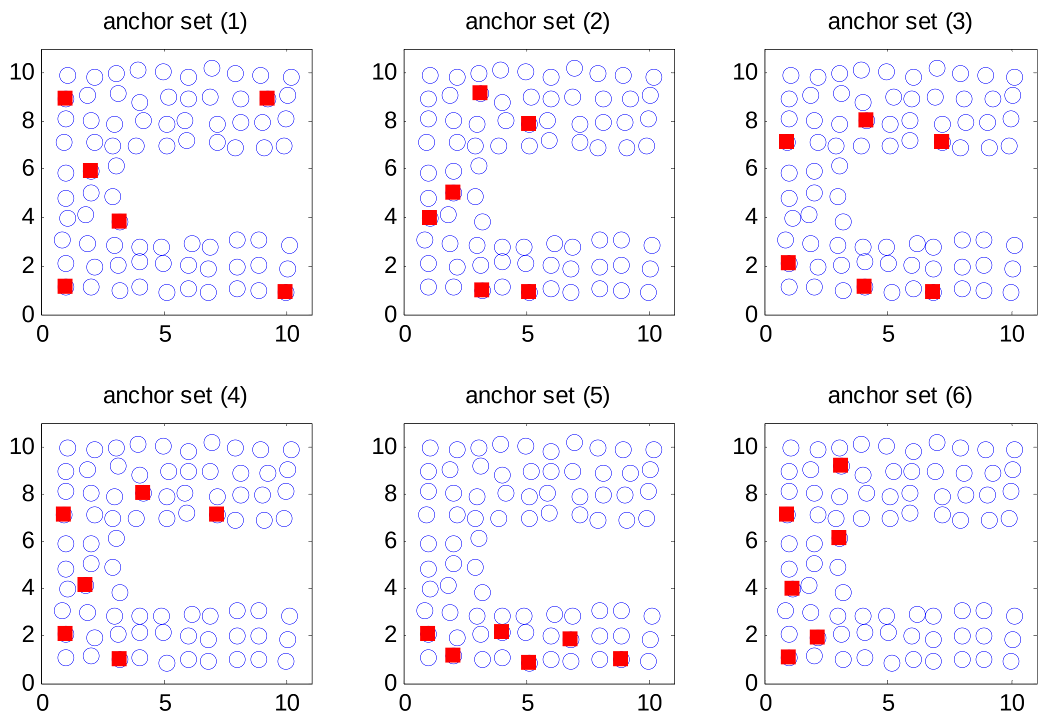

Figure 5 C-shape network scenarios with 6 anchor nodes

3.2.2 Experiment Results

In the experiments of C-Shape grid networks, the network average connectivity level increases from *CL=3.23* to *CL=13.25*, with radio range spanning from *r=1.2i* to *r=2.65i*. At the connectivity level lower than *3.23*, the network is often partitioned.

In range-based scenarios, with an average of *5%* measurement error on local distance between each pair of neighbouring nodes, the median errors of the estimated coordinates are illustrated in Figure 6, which depicts the average error on computed coordinates as a function of the average connectivity level. For any given number of anchor nodes, e.g., the three-anchor node cases, the

average (median) performance result is taken from all the scenarios that consist of the different three-node anchor sets as presented in Figure 3. It is also computed with several different starting nodes. Therefore, degraded positioning results, from poorly positioned anchor sets, are also taken into account. This is the same for all the other cases of different anchor nodes discussed in this report.

Both CCA-MAP and MDS-MAP perform fairly well in this network configuration. Employing CCA-MAP, at certain Connectivity Level (CL), *e.g.*,$CL>5.2$, using 3 anchor nodes, the median error can be contained to $error<0.1r$. MDS-MAP brings down the position median error to $error<0.1r$ when $CL>5.6$, though the error may increase for certain CL values.

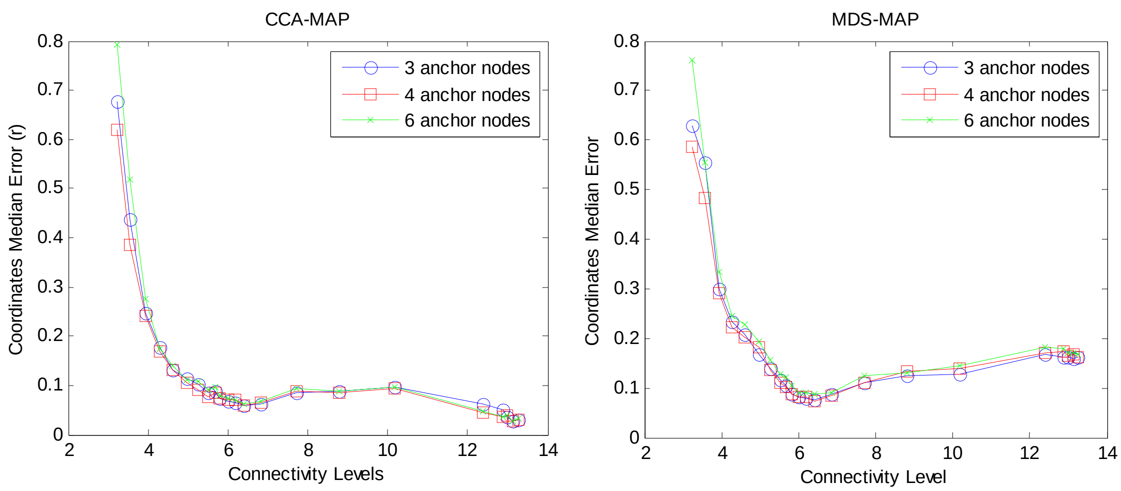

Figure 6 C-shape network of 79 nodes using range-based localization with 5% local distance measurement error

It should be noted that although almost all localization schemes reported could show improved performance when the sensor network has a high connectivity level, a large number of neighbouring nodes may not be plausible in real deployments as it hampers the radio network throughput. Therefore we are not particularly interested in using high connectivity levels to produce better positioning results but are more concerned with the performance at reasonable connectivity levels. Both MDS-MAP and CCA-MAP show the advantage of producing more accurate position estimates at relatively low connectivity levels compared with most of the existing methods. The CCA-MAP algorithm may be more stable at certain connectivity levels than the MDS-MAP in this network configuration.

Increasing the number of anchor nodes may improve the performance, though not necessarily. First, this is because any further improvement to the already quite accurate estimate results generated by CCA-MAP is difficult. Secondly, further improvement requires “good” positions of the anchor nodes, not merely more of them. In the experiments running both CCA-MAP and

MDS-MAP algorithms, for three anchor nodes, anchor sets (4) and (5), as illustrated in Figure 3, deliver the best position median accuracy. For four anchor nodes scenarios, anchor sets (4) and (5), as shown in Figure 4, again perform the best among all the five different sets. For the scenarios using six anchor sets, anchor set (4) (Figure 5) renders better than others. When the number of anchor nodes is increased from 3 to 6, the differences in the results from different sets of the anchor nodes are diminished. That is, when there are 6 anchor nodes in the network, most of the anchor sets deliver similar results except the set (5) as depicted in Figure 5, which is worse than the rest of the cases. For the scenarios of using only three anchor nodes, anchor set (4) and (5) perform apparently better than the other sets. Figure 7 depicts the performance of the different anchor sets for range-based scenarios. The cases of 6 anchors are not plotted here as most of the anchor sets deliver similar results except set (5) as described above. The anchor set numbers used in the figures and in the discussions here are the same as those illustrated in Figure 3, Figure 4 and Figure 5.

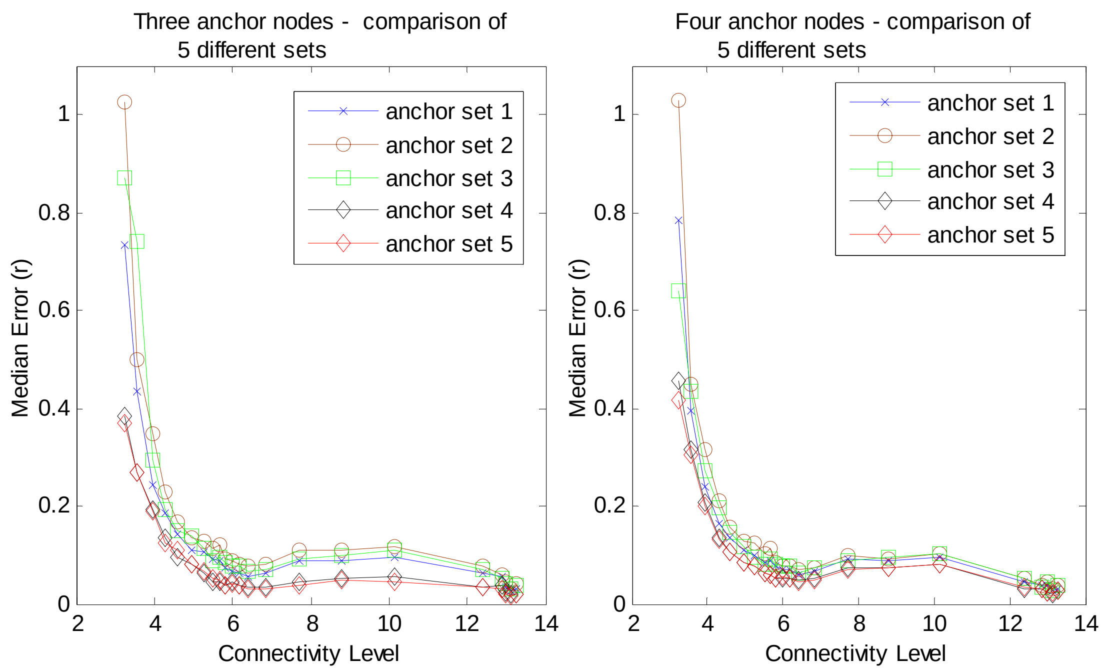

Figure 7 Performance of different anchor sets at different locations of the network

The comparisons between CCA-MAP and MDS-MAP for range-based scenarios are illustrated in Figure 8. CCA-MAP generates better results at certain ranges of the average node connectivity levels. CCA-MAP and MDS-MAP produce most similar results round *CL=6*, where CCA-MAP outperforms MDS-MAP by around *1%-2%r* in median position accuracy. When *CL<6* and *CL>7*, CCA-MAP improved MDS-MAP from about *3%r* to more than *10%r* in median position estimate accuracy.

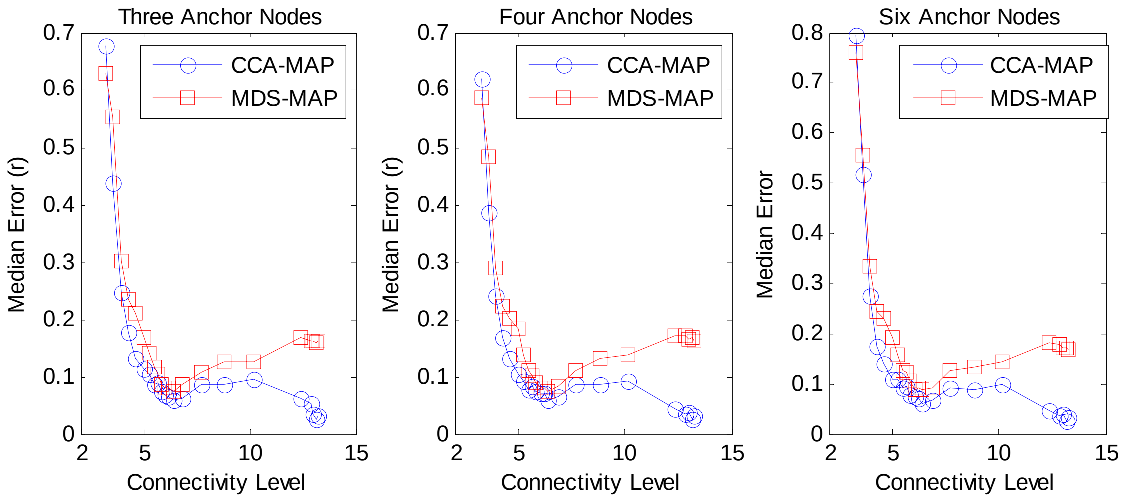

Figure 8 Comparison of CCA-MAP and MDS-MAP for range-based cases

Using only the connectivity information in range-free scenarios, the localization results for the same set of random networks are presented in Figure 9. Without ranging capabilities, the performance is degraded as expected. Using 3 anchor nodes, CCA-MAP achieves median error of position estimation at *0.14r* around connectivity level *CL=5.1*. Though the errors may fluctuate with increased *CL*, the median error stays at *error<0.2r* when *CL>5*. At *CL>12.86* the median error comes down to *error<0.10r*.

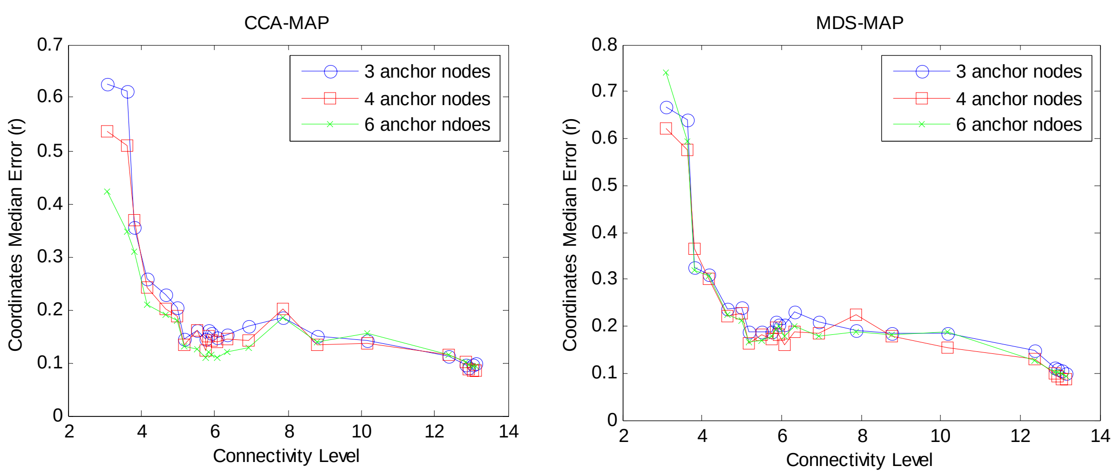

Figure 9 C-shape network of 79 nodes using range-free localization

Using the MDS-MAP algorithm with 3 anchor nodes, the median errors of position estimation come down to *error<0.2r* around connectivity level *CL=5.74*. Around *CL=13.14*, the median errors are brought down to *error<0.1r*.

Increasing the number of anchors improves the results especially for low connectivity levels. Note that various anchor sets for the six anchor scenarios were selected that include some badly positioned anchor sets (e.g., set (5) and (6) as shown in Figure 5). This may explain why the average median errors of the scenarios that encompass six anchor nodes do not perform considerably better than the scenarios that have only three or four anchor nodes. In fact, when using only a small number of anchor nodes, adding a couple of more anchor nodes may not help with the performance. The positions of the anchor nodes can impact the estimated results.

It can be observed that although a higher average node connectivity level does bring down the estimation errors, the results can fluctuate with the increased network connectivity level. With increased radio range and connectivity level, more nodes are included in the local map while more of them use the sum of the distances or hops over multiple hops to approximate their true distance. Thus when the radio radius increases from r to $r+\delta$, the approximated local distance matrix may be actually less accurate at $r+\delta$. However, a bigger matrix assists in the data reduction process to generate more accurate results due to the fact that more constraints are given in the matrix. These two factors combined together may result in a loss of accuracy when the matrix becomes too inaccurate and also the main trend of improved error ratios when the matrix is growing in size. This explains the local fluctuation of the results and the main trend of improvements with increased connectivity levels.

Note that in the published literature, the results reported often show no fluctuation but "well behaved" curves. This is because the authors did not compute the results using small paced values of connectivity levels and thus fail to observe the local swinging effect of the results.

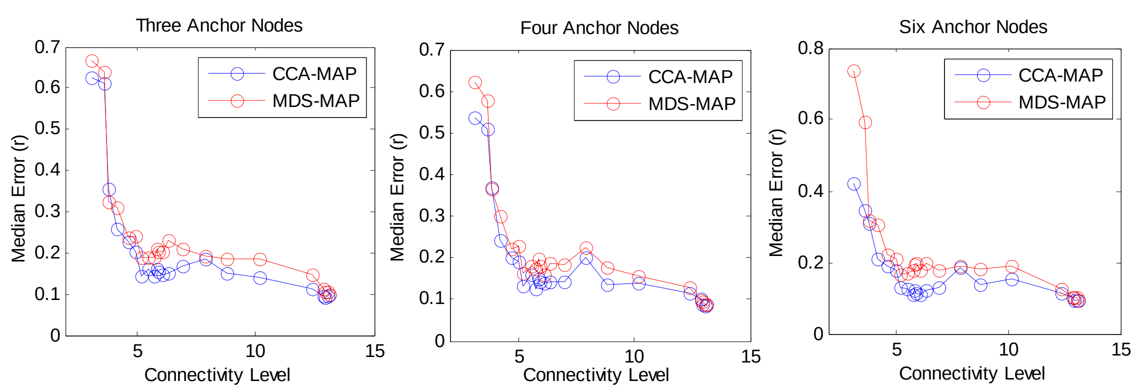

Figure 10 Comparisons of CCA-MAP and MDS-MAP for range free options

The comparisons between CCA-MAP and MDS-MAP for range-free scenarios are presented in Figure 10. CCA-MAP performs better than MDS-MAP for certain connectivity levels from *CL=5* to *CL=13* (about *3-4%r*). For *5<CL<7*, CCA-MAP improves MDS-MAP for about *4%r* to *7%r* most of the time. The difference between CCA-MAP and MDS-MAP decreases when the CL

increases. At *CL>12*, the different between CCA-MAP and MDS-MAP is diminished to about *1%r* and less.

In range-free localization scenarios, the results are less accurate and fluctuate more. When using *3* anchor nodes, CCA-MAP generates on average better results at *CL>5* using anchor set (4) shown in Figure 3. Other anchor sets perform comparably except set (2) which is often worse than others. With four anchor nodes, CCA-MAP range free localization favours anchor set (4) and (5) as shown in Figure 4. The four anchor node set (2) is still worse in its performance compared with other sets. When using 6 anchor nodes, anchor set (4) as depicted in Figure 5 performs better, while anchor set (5) is ranked the worst among the six anchor sets. Anchor set (1) of six anchor nodes does not perform as well as others either.

For MDS-MAP range free algorithms, when using three anchor nodes, anchor set (4) on average performs better than other sets. Anchor set (2) and (3) do not perform as well as other sets. Using four anchor nodes, anchor set (5), as show in Figure 4, generates on average better results. Anchor set (3) is worse than other sets most of the time. When using 6 anchor nodes, anchor set (5) is unstable in generating the results and performs considerably worse than other sets. Set (1) is also slightly worse (from *2%-6%r* worse when CL>5) than other sets.

When merging local maps into a global map, selecting a different local map as the starting map for the merge process does not seem to make any consistent significant difference in the results.

3.3 Rectangle Networks

3.3.1 Network Experiment Parameters & Configurations

The network configurations in this subsection form a coverage area in the shape close to a rectangle. Such a long shaped sensor network may be deployed for monitoring roadside or other approach areas. In the experiment, we select a network area consisting of a $25r \times 4r$ rectangle to deploy 100 nodes to grid points with node placement error uniformed distributed in the range of *[-20%i, 20%i]*, taking *i=1* as placement unit. Figure 11 depicts such an example network.

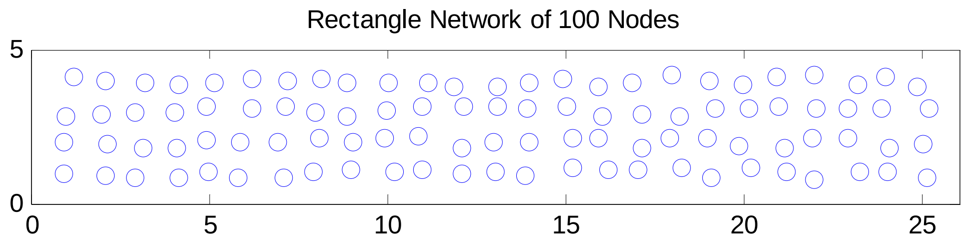

Figure 11 A Rectangle network of 100 nodes

Different anchor sets of different positions were selected for the localization simulations. In Figure 12, Figure 13 and Figure 14, network configurations with three, four and six anchor nodes

deployed to various locations are presented. As in the previous subsection for C-shape networks, all these cases, consisting of various anchor node positions, are computed to report their average performance as well as their performance deviations.

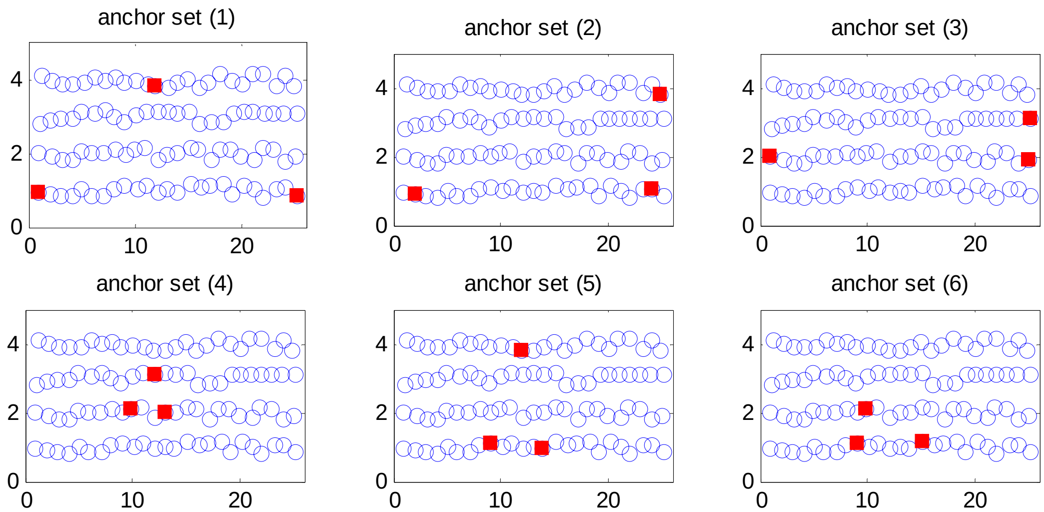

Figure 12 Rectangle network scenarios with three anchor nodes

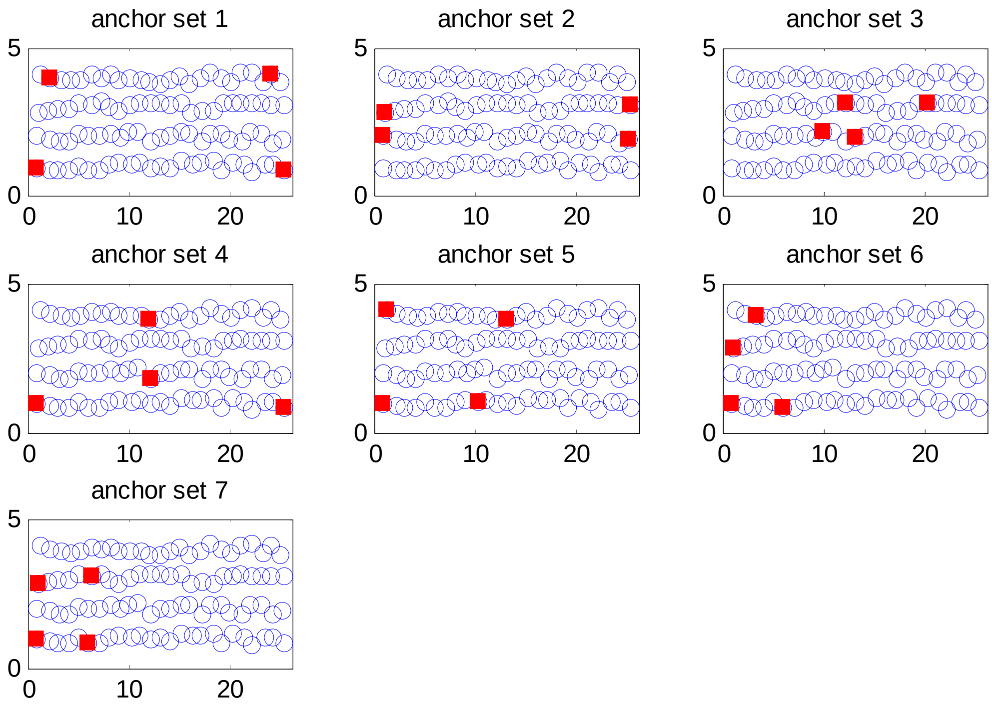

Figure 13 Rectangle network scenarios with four anchor nodes

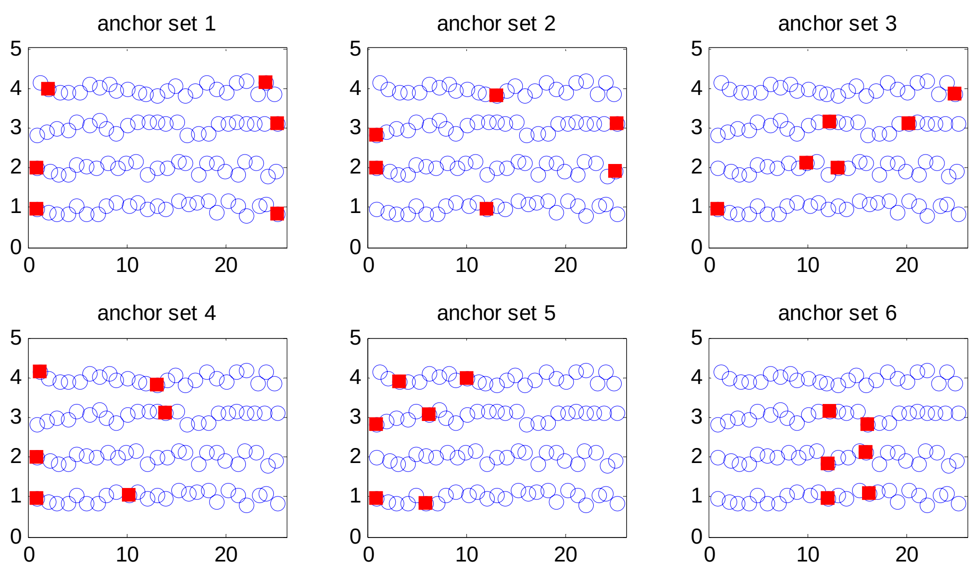

Figure 14 Rectangle network scenarios with six anchor nodes

Please note that the diagrams in the figures (Figure 12, Figure 13 and Figure 14) were plotted to show the positions of the anchor nodes. Due to the space limitations, the sensor nodes look to be crowded together, though in the networks, they are well paced to form the coverage as described above. Please note that there is no corresponding relationship between the set numbering in scenarios consisting of different numbers of anchor nodes. For example, set (3) of three anchor nodes and set (3) of four anchor nodes may have no commonality in the anchor node positions. Thus, in the following discussions, one should not assume that set (3) of four anchor nodes perform better than set (3) of three anchor nodes.

3.3.2 Experiment Results

In the experiments of rectangle networks, the network average connectivity level increases from *CL=3.26* to *CL=14.12*, with radio range spanning from *r=1.2i* to *r=2.65i*.

In range-based scenarios, with an average of *5%* measurement error on local distance between each pair of neighbouring nodes, the median errors of the estimated coordinates are illustrated in Figure 15 which depicts the average error on computed coordinates as a function of the average connectivity level.

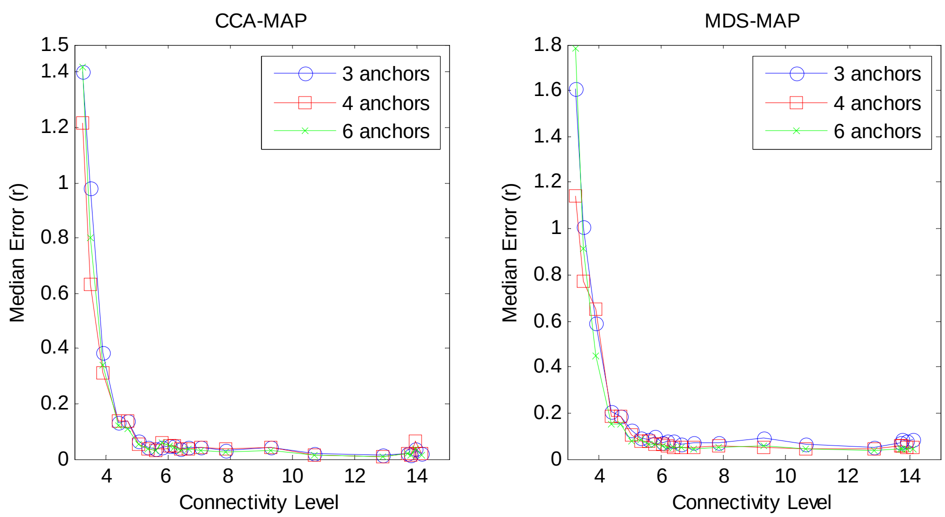

Figure 15 Rectangle 100 node-network using range-based localization with 5% local distance measurement error

Using CCA-MAP, at Connectivity Level *CL>5.1*, with 3 anchor nodes, the median error can be contained to *error<0.1r*. When *CL>5.38*, using only 3 anchor nodes, CCA-MAP can push down the average position median error to *error<0.05r*. MDS-MAP as well brings down the position median error to *error<0.1r* when *CL>5.1*, though the error remains between *0.05r<error<0.1r*. Overall, using the range based method, both CCA-MAP and MDS-MAP perform well for this network configuration.

When using 3 anchor nodes in the network, in both CCA-MAP and MDS-MAP schemes, anchor set (1) as shown in Figure 12 on average performs the best, especially when *CL>7*. Anchor set (4), (5) and (6) generate less accurate results than others when *CL>5* especially when using MDS-MAP. For example, using MDS-MAP anchor set (4) may produce *10%r* more median errors on estimated coordinates than anchor set (1) when *CL>8*. With the CCA-MAP algorithm, the difference between anchor set (4), (5) and the rest of the sets is not significant, e.g., about *1-2%r*. From *CL=5* to *CL=7*, using 3 anchor nodes, anchor set 6 may generate reasonably good results for CCA-MAP as well. The comparisons of the results produced by different anchor sets are not plotted here because most of them are not significant and the differences hard to be depicted well. It should be understood that the average results here only demonstrate the average trend. For a specific deployment case, the particular topology will affect the results and should be simulated for evaluation. It is generally true that at lower connectivity levels, the anchor positions are more important to maintain a low error ratio.

Deploying four anchor nodes in the network, we tested some outlier positions for the anchor nodes so as to cluster them at only one side of the network, as shown in Figure 13. Using MDS-MAP, anchor set (4) performs better while anchor set (6) and (7) are worse than others. The difference between anchor set (6) and anchor set (4) can be more than *20%r* when *CL<7* and about *2%-7%r* when *CL>7*. Using CCA-MAP, in addition to anchor set (4), set (3) also performs well. Anchor set (6) and (7) are ranked worst. Here the difference between anchor set (6) and anchor set (4) can be more than *20%r* when *CL<7* and about *1%-5%r* when *CL>7*.

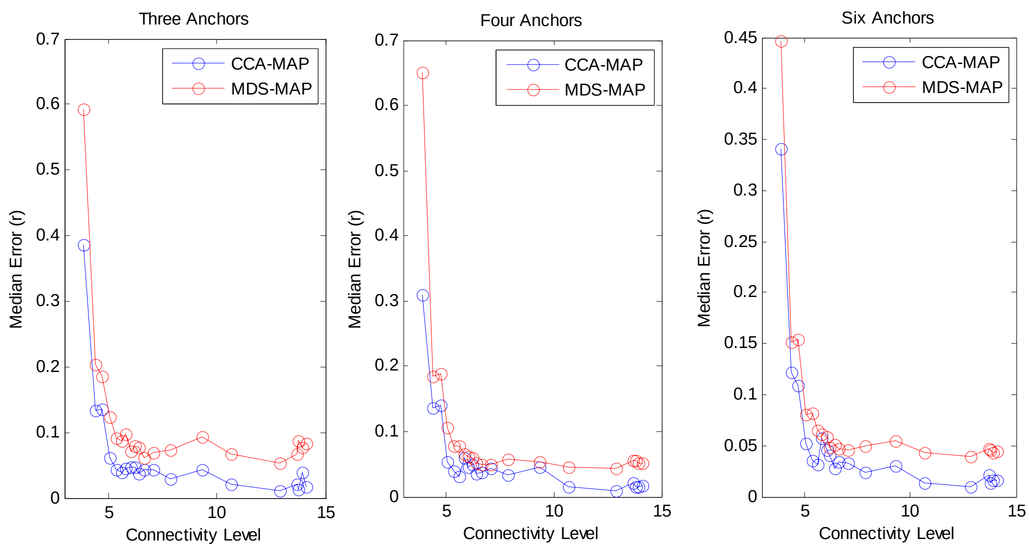

Figure 16 Comparison between CCA-MAP and MDS-MAP for rectangle network of 100 nodes: local distance measured with 5% error

Having six anchor nodes in the network, CCA-MAP does not generate major differences in the results obtained using different anchor sets. When *4<CL<6*, set (5) and (6) may not perform well (about *4-7%r* worse). When *CL>6*, set (5) and (1) may have a few "bad" points, though the

difference is not significant (about *2%r*). MDS-MAP scheme in this case also ranks set (5) as the worst in average (about *2%r* worse). At *6<CL<8*, anchor set (1) may have a few "bad" points of about *3%r* worse than set (3) and set (4), while set (6) may generate *3%-4%r* more error ratio at *CL>13*.

Figure 16 illustrates the comparisons between CCA-MAP and MDS-MAP for the rectangle network scenario in range-based scenarios. Compared with MDS-MAP, CCA-MAP may deliver about *4%-5%r* accuracy improvements when *CL>4*.

The performance results from the range-free method using both CCA-MAP and MDS-MAP are presented in Figure 17. Without ranging information, rectangle networks become challenging for MDS-MAP. Though it may achieve a median error ratio *<20%r* for some connectivity levels, the results fluctuate bouncing around *20%-40%r* or even higher. The results from CCA-MAP also fluctuate with median error between *10%-30%r*, mostly under *20%r*, from *CL=5.08* to *CL=12.88*. At *CL>12.88*, the median error ratio is contained mostly as *error< 10%r*.

The long shape of the rectangle network is more challenging than square shaped networks for distributed map algorithms. One reason is because of the special grid like structure where the nodes lying at diagonally adjacent intersections always bring errors when adding hop distances to approximate the true distance between the two nodes. This is actually true for most grid types of network. Additionally, for rectangle networks, the linear topology adds up the distance matrix errors along certain directions rather than possibly cancel them as in a more regular square type shape that expands to all directions.

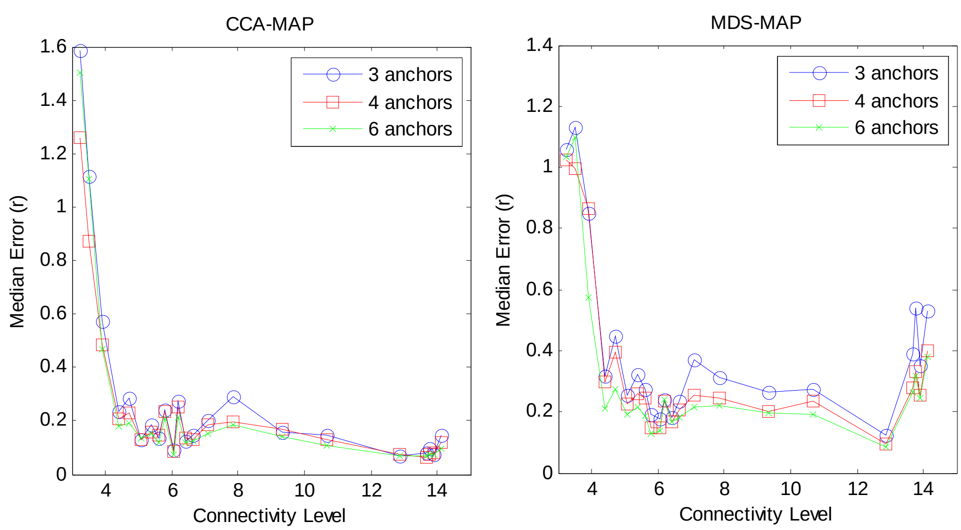

Figure 17 Rectangle network localization using connectivity information only

Figure 18 illustrates the differences between CCA-MAP and MDS-MAP in range-free scenarios. CCA-MAP can outperform MDS-MAP for about *10-20%r* in certain CL ranges and about *3-8%r* around *5<CL<7*.

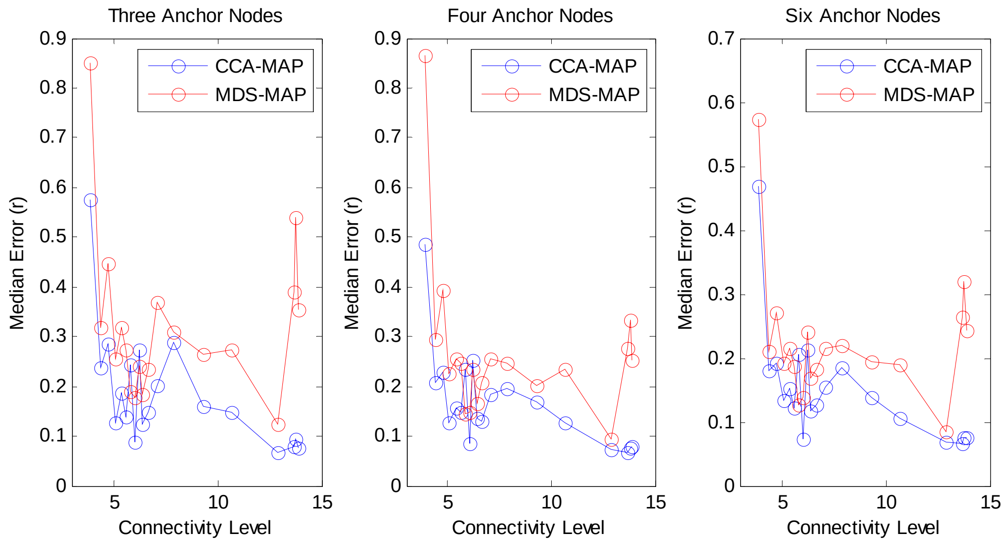

Figure 18 Comparison of CCA-MAP and MDS-MAP: rectangle network & range-free

In range-free scenarios, when using only 3 anchor nodes, anchor set (1) performs well at *CL>5* in both CCA-MAP and MDS-MAP algorithms, while set (4) is worse than other sets most of the time. The difference between results from set (1) and set (4) can be *20%*r at certain CL. Deploying 4 anchor nodes as shown in Figure 13 for both CCA-MAP and MDS-MAP, anchor set (4) and also (3) perform relatively better and are more stable than other sets at *CL>4*. Set (6) and set (7) are on average the worst in the performance ranks. In the scenarios of 6 anchor nodes, on average anchor set (4) performs better than other sets. Using 6 anchor nodes, anchor sets (5) and (6) are least accurate. Anchor set (1) also can have bad spike error points. In the scenarios, the difference between results generated by anchor set (4) and a less accurate anchor set, set (5), for example, can be more than *10%r*.

3.4 Area Surveillance Networks

3.4.1 Network Experiment Parameters & Configurations

We selected an irregular type of network that is close to the shape of a rectangle but not strictly a rectangle. This type of network intends to model the SASNet scenarios of road junction or choke point monitoring. Figure 19 below plots a sample network of this type where nodes are spaced but with jitters assuming terrain obstacles or other conditions. Though both CCA-MAP and MDS-MAP were tested before [10][4] showing good performance for squared grid type networks, the network not strictly square in shape, nor really grid in node placement may be closer to the real scenario.

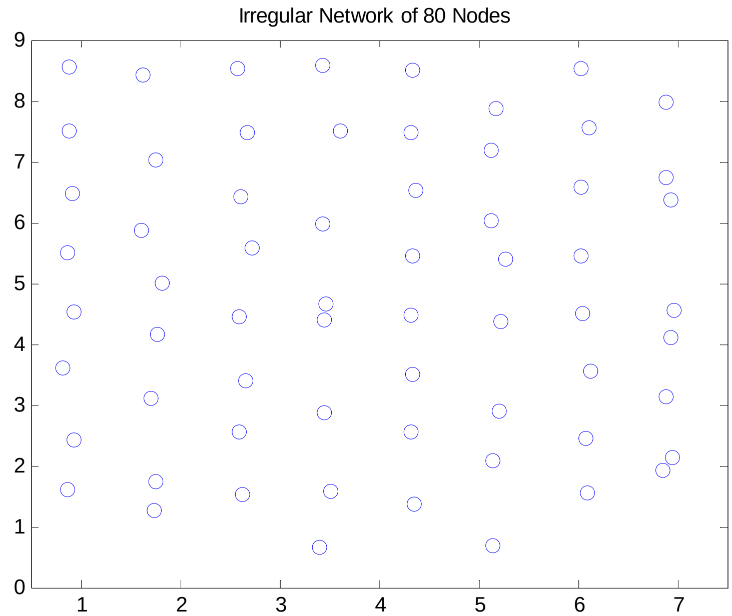

Figure 19 An Irregular Network of 80 Nodes

Figure 20 and Figure 21illustrate the scenarios where each consists of three or four anchor nodes deployed at different positions. As for the previous network types, CCA-MAP and MDS-MAP algorithms are applied to compute location estimates for each of these scenarios across multiple network samples to obtain the performance measurements.

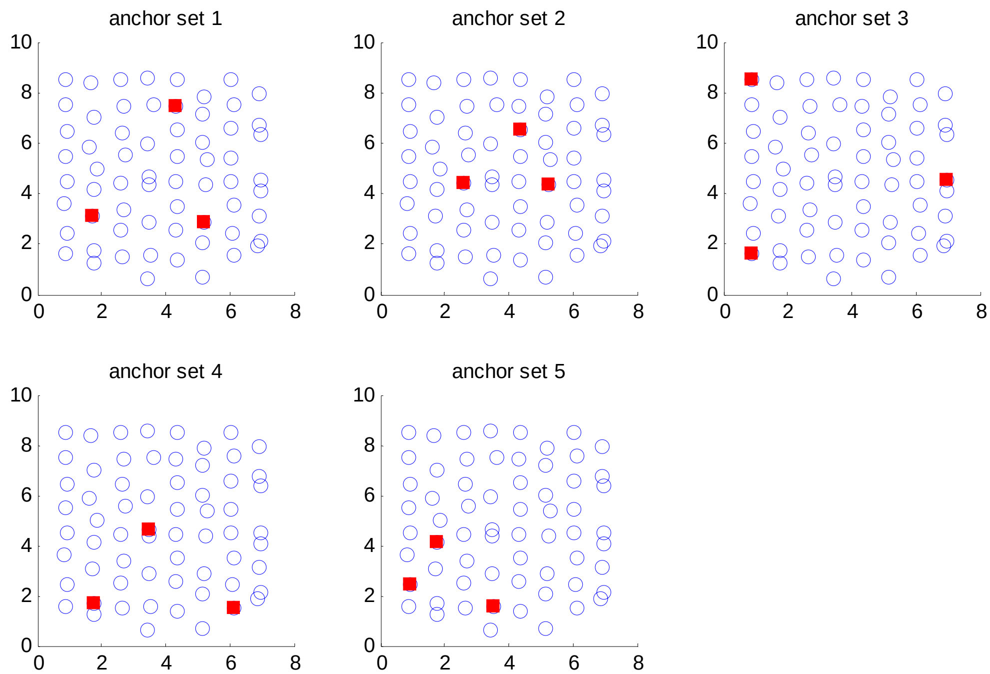

Figure 20 Irregular network scenarios with three anchor nodes

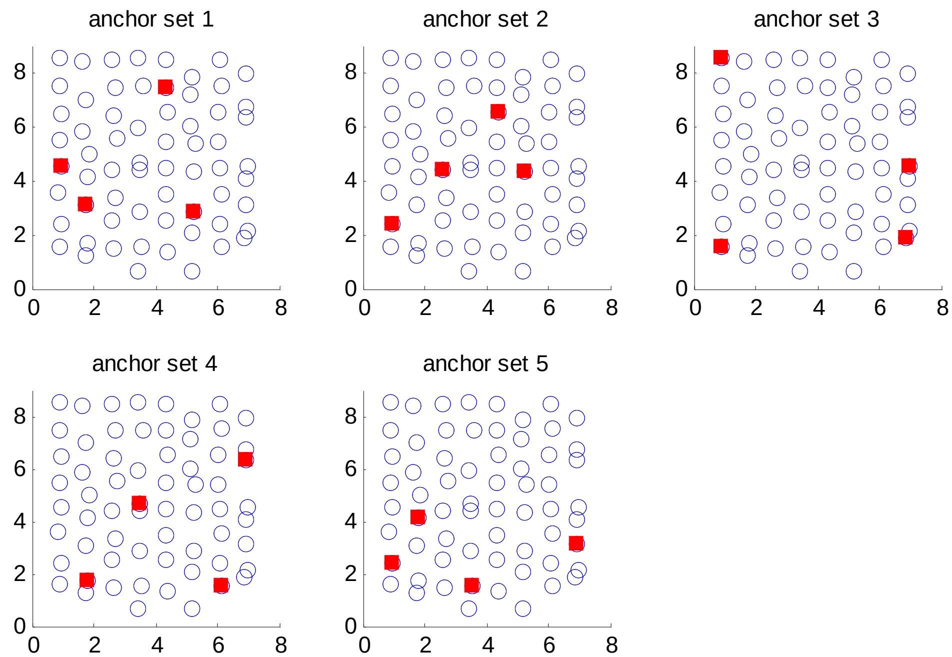

Figure 21 Irregular network scenarios with four anchor nodes

3.4.2 Experiment Results

In the experiments of irregular networks, the network average connectivity level increases from *CL=4.28* to *CL=17*, with radio range spanning from *r=1.2i* to *r=2.65i*.

Knowing local distance with *5%r* average measurement error, both CCA-MAP and MDS-MAP algorithms deliver good position accuracy for this type of irregular networks. Figure 22 below illustrates the median error of the estimated node coordinates using only three anchor nodes. Applying CCA-MAP, at *CL>4.43*, the median error is contained within *10%r*. At *CL>5*, the median error is brought down to *<5%r*. At *CL>10*, the median error is under *1%r*. The results generated by MDS-MAP are about *1%-2%r* less accurate than those generated by CCA-MAP. We do not display plots of results generated by using more anchor nodes as the accuracy level produced by only three anchor nodes is quite satisfactory.

Among the different 3 anchor sets as shown in Figure 20, anchor set (1) performs well in both CCA-MAP and MDS-MAP algorithms using the range-based method, as illustrated in Figure 23. Anchor set 5 appears to be the least accurate set in both CCA-MAP and MDS-MAP computations. Using CCA-MAP, anchor set 4 may have a more stable performance than using MDS-MAP. However, the differences are mostly not significant being around only *1%-2%r*.

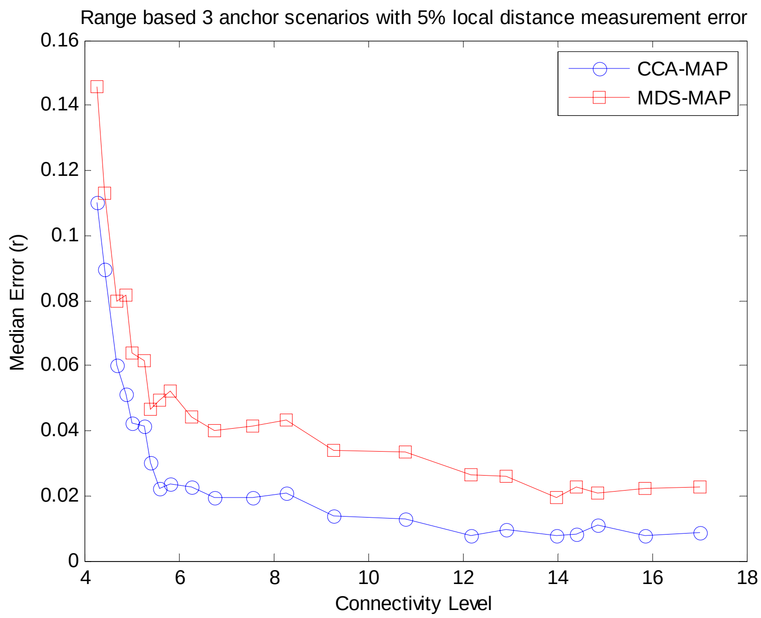

Figure 22 Irregular networks with 80 nodes: range-based options

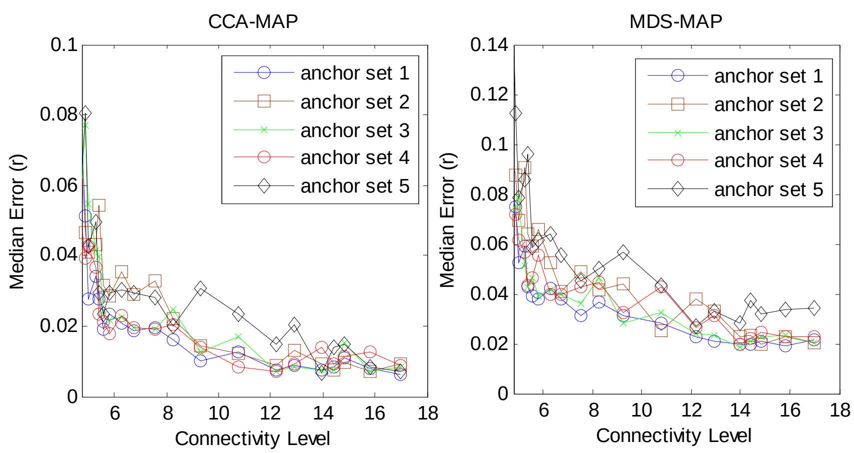

Figure 23 Performance for irregular network works using range-based method with 3 anchor nodes

Without range measurements, CCA-MAP and MDS-MAP project the node position coordinates using only the connectivity information. The accuracy of the estimates is deteriorated as expected. The results also fluctuate along different node connectivity levels. Employing CCA-MAP, the position median error may be brought down to around *10%r* when *CL>6.28*. When *CL>8.2*, the median error is consistently under *10%r* approaching *5%r*.

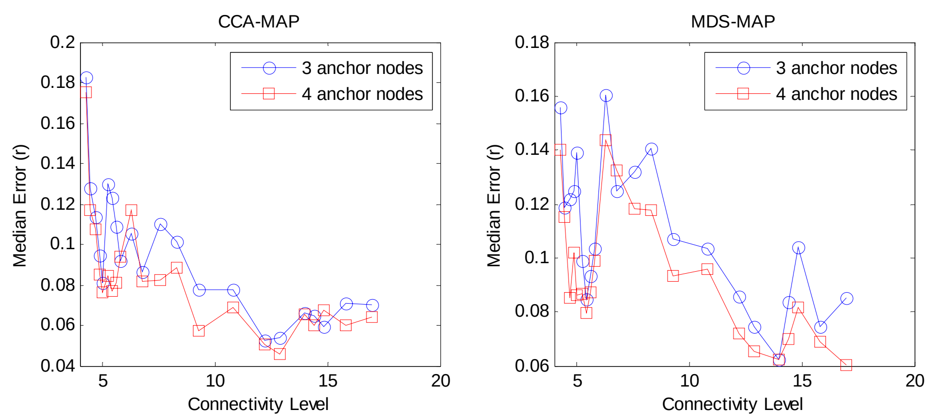

Figure 24 Performance of Irregular networks: range-free cases

Applying the MDS-MAP algorithm, the results fluctuate more with the increased connectivity levels, especially when the node average connectivity is at $5.8<CL<12.9$. Results from the MDS-MAP algorithm are about *2%-4%r* less accurate than those generated by CCA-MAP when $5.8<CL<12.9$, as shown in Figure 25.

Comparing different sets of anchor nodes, in range free scenarios, anchor set (1) in Figure 20 of three anchors and set (1) in Figure 21 of four anchors perform relatively consistently and well across different connectivity levels. Other anchor sets all have spikes of error surges at certain connectivity levels. Increasing the number of anchor nodes or the network connectivity level mitigates the differences between the results generated by different anchor nodes.

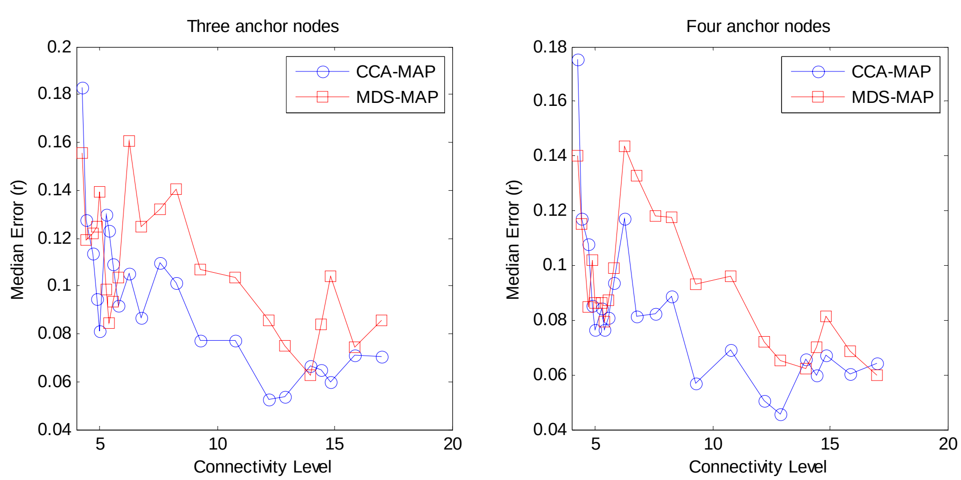

Figure 25 Performance comparison of CCA-MAP and MDS-MAP for range free cases

3.5 Random Networks

3.5.1 Network Experiment Parameters & Configurations

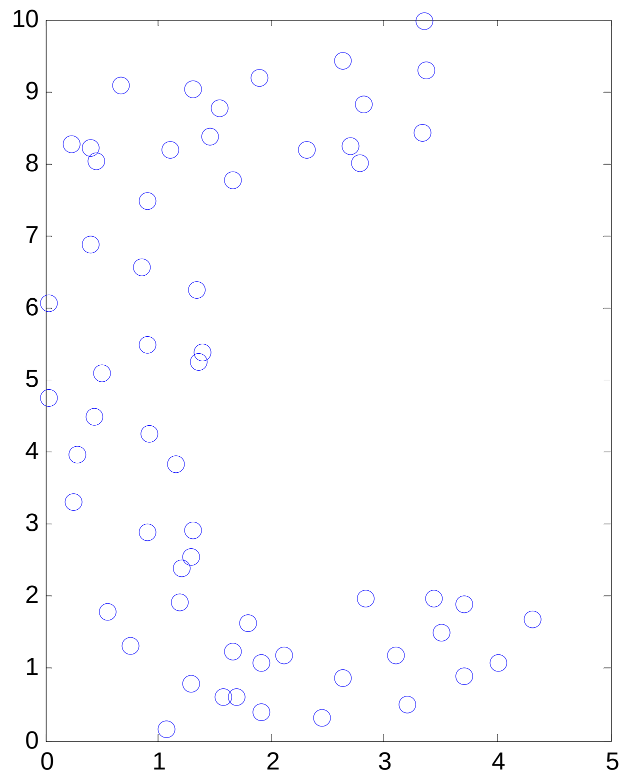

Figure 26 Random partial loop network of 60 nodes

Though random networks may not be practical in SASNet field deployment, we still cover it briefly here to present a more complete set of different network scenarios. More thorough studies on localization of nodes in random network configurations applying MDS-MAP and CCA-MAP can be found in [10]][4]]. We select in particular the network configuration of a partial loop as show in Figure 26. Different from the C-shape type of network, the partial loop network takes a longer shape rather than a complete symmetric square area. This places extra challenge in addition to the fact that the nodes are randomly placed. Figure 27, Figure 28 and Figure 29 illustrate the scenarios of different anchor sets selected for the experiments.

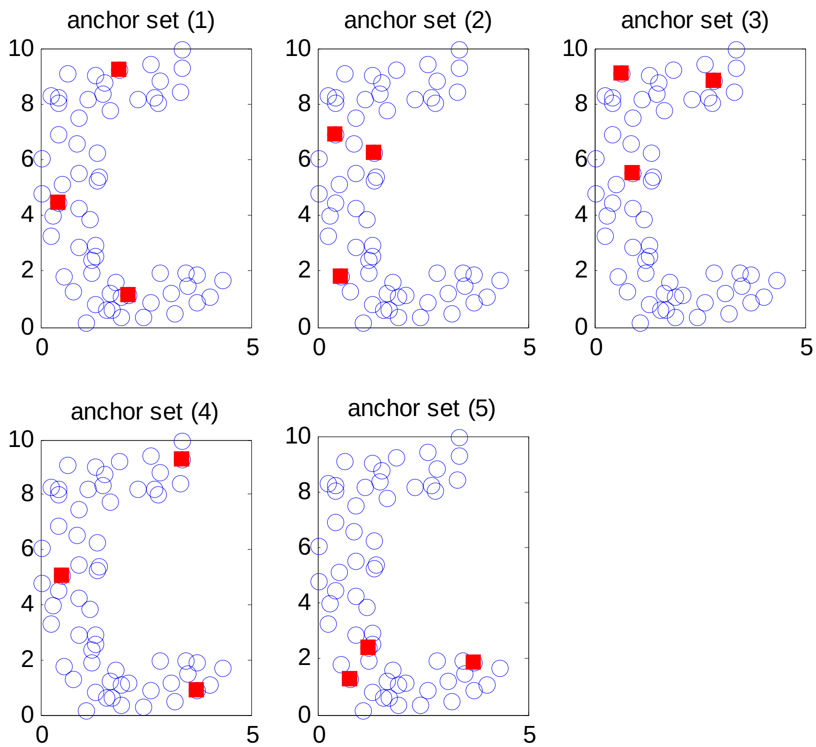

Figure 27 Partial loop network of 60 nodes: 3 anchor scenarios

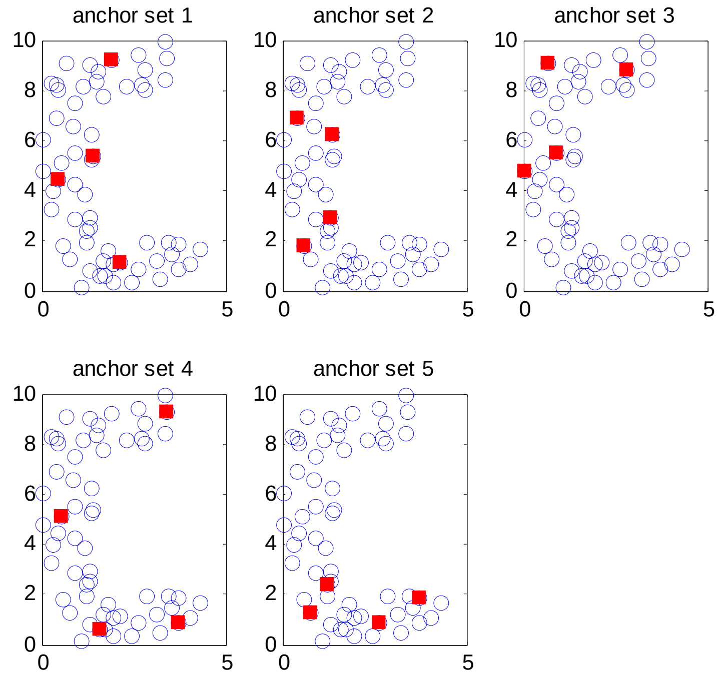

Figure 28 Partial loop network of 60 nodes: 4 anchor scenarios

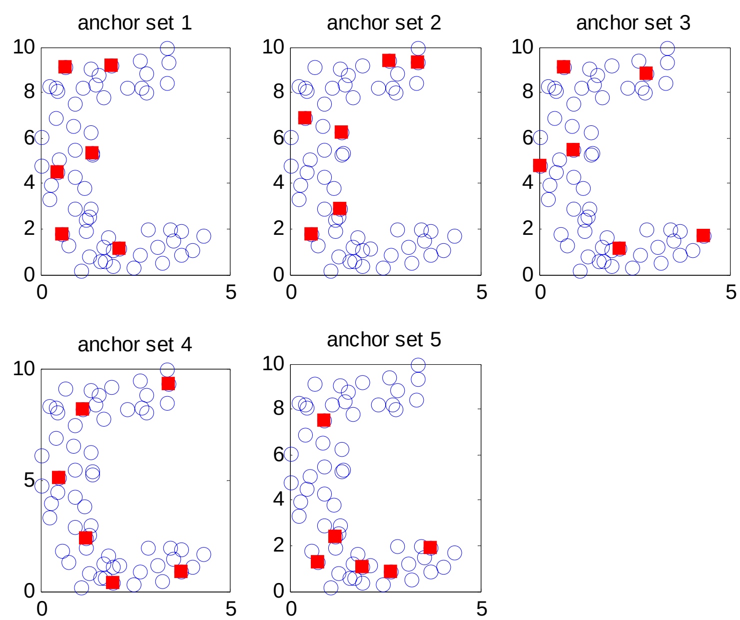

Figure 29 Partial loop network of 60 nodes:6 anchor scenarios

3.5.2 Experiment Results

In the experiments of partial loop networks, the network average connectivity level increases from *CL=5.27* to *CL=21*, with radio radius spanning from *r=1.0i to r=2.65i*.

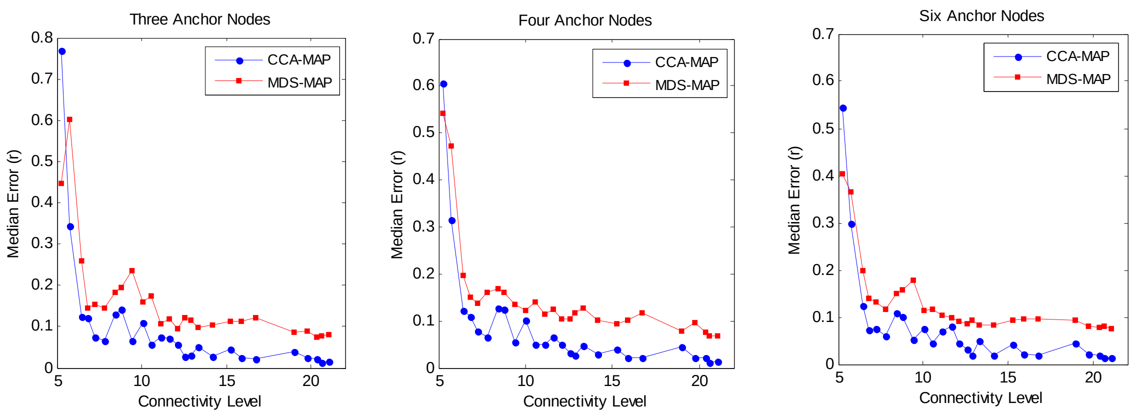

Figure 30 Range based localization with 5% of local distance measurement error

Knowing local distance with *5%* of measurement error, the location estimation results are presented in Figure 30. The CCA-MAP algorithm can contain the position estimate error within *10%r* at *CL>10* and within about *5%r* at *CL>12.56* using *3* anchor nodes. The MDS-MAP algorithm can also bring the median error down to *<10%r* at *CL>16*. Using 3 and 4 anchor nodes in CCA-MAP, anchor sets (1) and (4) perform better at lower *CL (CL<8)* region, while set (5) is the worst. In MDS-MAP, set (1) performs better than other sets. Set (4) does not perform as well as in CCA-MAP.

Without any ranging capabilities, both CCA-MAP and MDS-MAP have difficulties in delivering the estimation accuracy under*10%r*. The results are presented in Figure 31. The results from both algorithms are fluctuating and unstable. Using 3 anchor nodes, CCA-MAP generates position estimates with about *30%r* median error. Deploying 6 anchor nodes, both MDS-MAP and CCA-MAP produce results of having median errors at about *25-30%r*.

For partial loop random networks, if using only connectivity information, neither CCA-MAP nor MDS-MAP delivers the required position estimation accuracy. In fact, even for the uniform square shaped random networks, it was found in our previous studies [4]][5]] that the range free CCA-MAP and MDS-MAP method could hardly bring the median error down to *<10%r* when using *3-5* anchor nodes at *CL<20*.

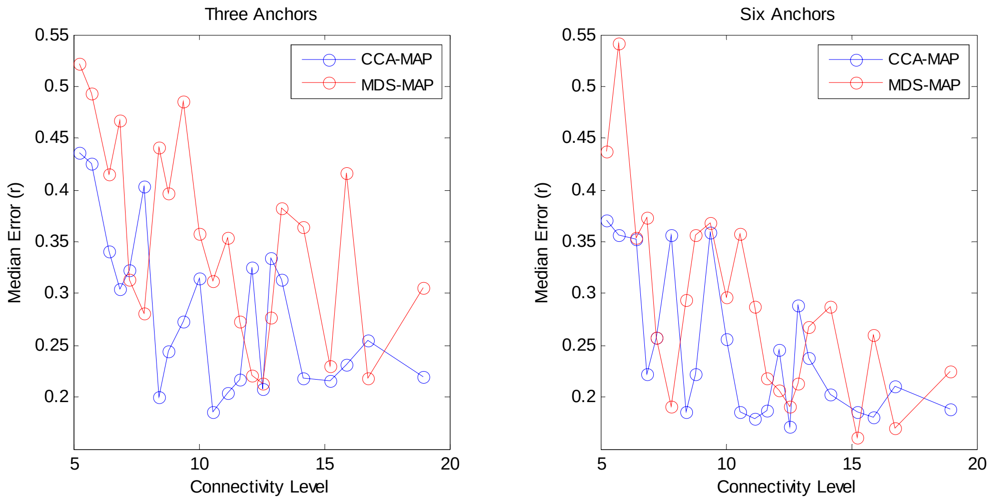

Figure 31 Partial loop random network: range-free cases

3.6 Issues and Analysis

3.6.1 Parameters and Computing Cost

As demonstrated in the previous subsections, accurate position estimation results may be achieved using CCA-MAP and MDS-MAP for many different scenarios using only a minimum number of anchor nodes. However, the computational cost of such cooperative algorithms is a major concern requiring further investigations and careful assessment.

The following observations were made through the experiments of the two different map algorithms on various network scenarios.

CCA-MAP may achieve better accuracy than MDS-MAP in mapping node positions for many different scenarios. The CCA algorithm used in CCA-MAP has a much stronger minimization process explicitly targeting the cost function of the distance matrix, compared with the classic MDS algorithm used in MDS-MAP which adopts the eigenvector approximation of coordinates. Thus, CCA-MAP can be more accurate than MDS-MAP.

Though the CCA reduction algorithm has a smaller computational cost than the MDS reduction, ($O(N^2)$ vs. $O(N^3)$), there is hardly much difference in the computing complexity of local maps as the local maps are all very small. In fact, as the MDS solution has a closed form and CCA does not, CCA may be made to run multiple times to search for a better point projection because every run can deliver a different set of outputs. This benefits the algorithm in attaining better mapping accuracy. The disadvantage is that retry increases considerably the computation time. The small size of the local map distance matrix also requests a relatively large cycle value for CCA (e.g.,

100 was used in our experiments). It is expected that CCA local map computation would not be much lighter than MDS local map computing, if not "heavier". The refinement process used in MDS-MAP would exceed the computing expense of CCA-MAP as the refinement process is a minimization process which is less efficient than CCA. The refinement process also needs to be applied on the global map to produce better results, which is very expensive due to the size of the global maps. CCA is only applied to the local map thus has no cost surge for a large network size.

We did not observe significant improvement brought by the refinement process in the MDS-MAP algorithm for the network configurations and the connectivity levels of our interest.

The execution time of local maps using CCA-MAP varies for different scenarios. The real factors influencing the execution cost come from the input data matrix of the approximated distance matrix of local maps. A smaller accurate distance matrix would take less time to compute than a bigger inaccurate distance matrix. If too small, the distance matrix makes it harder to map accurately the output points. Then it may also increase the computational expenses with increased loop cycles. The matrix size and the cycle values can be tuned for CCA algorithm to handle different network configurations using the most suitable settings.

Due to the poor performance of “loop” executions in Matlab, the time benchmark of CCA algorithms cannot reflect the real time expense should it be implemented in a real system. During the experiments, using the range-based option, the average time for computing each local map using CCA-MAP ranges from *0.2sec* to *1 sec*, except in the irregular network scenarios where the local map computing time averages between *0.5sec* to *1.5sec*. Increasing the connectivity level thus the size of the local distance matrix would extend the computing time temporarily for a few CL levels but not always. This is because when the connectivity level is higher the distance matrix will be bigger and more accurate which in turn speeds up the CCA mapping process. So at higher CL, the average computing time tends to decrease.

In range-free cases, the approximated distance matrix is much less accurate and thus extends sharply the time of CCA mapping. In average, it takes from *0.2sec* to *3.7sec* to compute each local map. Increasing the connectivity level in range-free cases almost always prolongs the execution time, as the bigger matrix here is usually not more accurate.

In the current implementation of CCA-MAP and MDS-MAP algorithms in Matlab, MDS-MAP can execute much faster than CCA-MAP because of the poor loop execution performance encountered by CCA-MAP. "Loop" execution in Matlab has extremely poor performance, thus "loops should be avoided at all cost" in Matlab programming. MDS-MAP implementation has avoided using loops by invoking the eigenvector/eigenvalue decomposition function offered by Matlab. CCA-MAP on the other hand runs in cycles to minimize the cost function. CCA uses the results from the previous loop to compute the results for the current loop. Due to its sequential computing process, we have not yet been able to vectorize the CCA algorithm to avoid using loops in Matlab. In fact, the CCA algorithm is about loops. Therefore, CCA execution in Matlab may be much slower than if using a compiler based language on a real node platform.

The time batch marking obtained from Matlab simulation cannot well serve to determine the cost of CCA-MAP execution in a platform using a real implementation programming language such as C. One of our next steps is to port CCA-MAP to C to identify the execution cost of CCA-MAP.

In the next section, we will further analyze the implementation requirements and the computational costs comparing between CCA-MAP and MDS-MAP schemes.

3.6.2 Effects of Range Errors

Increasing the range errors often degrades the position estimation results. Larger range error in local distance measurements causes the distance matrix of the local map to lose its accuracy. The sensitivity of a localization scheme to various range errors should be understood. In the experiments presented so far, the range measured between neighbouring nodes is modeled as a random value drawn from a normal distribution with actual distance as the mean and a standard deviation of *5%*. Here, we increase the error deviation from *5%* to *50%*. We took the rectangle network of size 25×4 deployed with 100 nodes. Figure 32 shows the effects of increased range errors on the localization results produced by CCA-MAP, compared with those by MDS-MAP. When the range error increases, so do the errors on the node position estimates generated by both methods. In general CCA-MAP sustains its performance advantage under the increased range errors compared with MDS-MAP. This is because that CCA in its non-linear unfolding and mapping process has a tendency to converge towards the real distance matrix, even though the real distance matrix is unknown. When the range error becomes very large, e.g., more than 50%, the accuracy of the CCA-MAP algorithm may reduce sharply as CCA-MAP in general prefers an accurate distance matrix. MDS-MAP on the other hand may tolerate a vastly erroneous distance matrix better than CCA-MAP, as MDS approximates the positions rather than optimizing precisely to the distance.

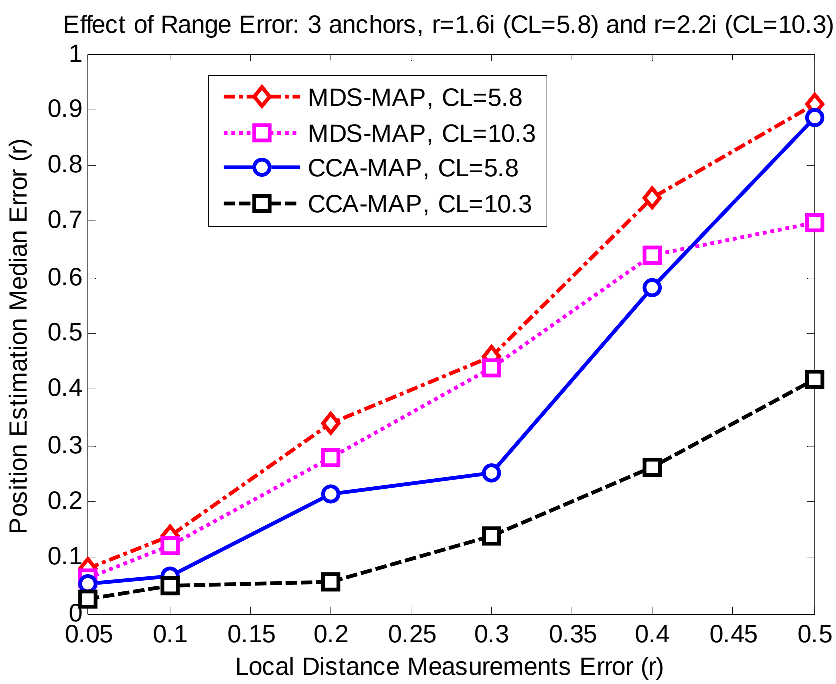

Figure 32 Effects of range errors on the position estimations of rectangle network

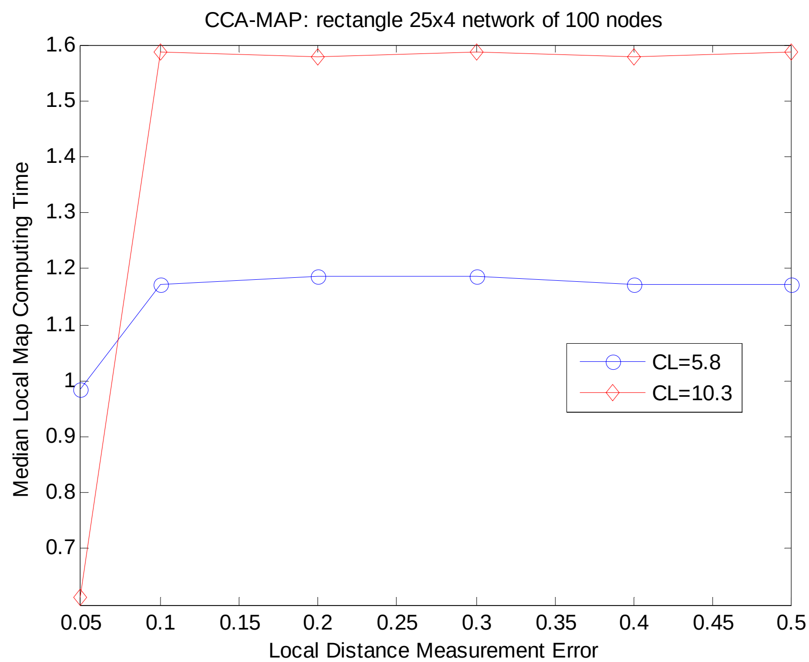

Figure 33 Error effects on local map computing time of CCA-MAP

For CCA-MAP, a less accurate distance matrix may increase the local map computing time. The computing time for a local map is illustrated in Figure 33 when the range error increases from *5%* to *50%*. It can be seen that the computing time surges when error variance is at *10%*. Then the computing time stabilizes without much further increase.

4 Design Considerations for SASNet Localization Scheme

In addition to the localization algorithm, which is the focus of this report, the implementation of a localization scheme involves many other issues. These issues are addressed briefly in this section to capture the options that we currently have.

4.1 Distribution of Algorithm Implementation

The first issue we describe here is the location for executing the localization scheme. Different localization algorithms would prefer different places to carry out the computations employing the required protocols and messaging among the nodes involved.

The cooperative algorithms, including MDS-MAP and CCA-MAP that have been discussed here, is quite different from other iterative algorithms as reviewed in Section 2. As indicated there, iterative algorithms are naturally distributed and intended to execute on each of the sensor nodes. In iterative schemes, through message propagation, the locations are calculated over each node across the network. The cooperative algorithm often computes the map of the subnetwork or of even the entire network. Thus the cooperative algorithm can be carried out on gateway nodes. Though cooperative algorithms may also be executed in a fully distributed manner on each sensor node to compute each local map, and to merge the local maps at each different node [45], the computational cost may be of concern. We will further look into the computational cost in terms of memory and execution in the next subsection. The distribution of the localization scheme and the required messaging are examined here first. The messaging requirements are the same for the MDS-MAP and CCA-MAP algorithms.

4.1.1 Cooperative Localization Scheme on FN

SASNet exhibits the two tiered network architecture as shown in Figure 1. The FN (Fusion Node) has more computing and communication resources, and also maintains a fairly complete view of all the SN (Sensor Nodes) under its domain including their location information; it may be thus a viable option to implement localization computation completely or partially on FN to relieve the severely resource limited SN from the overhead of location estimation. We call this approach here the distributed gateway implementation. In Figure 34 such an approach to implement the cooperative localization in SASNet is depicted.

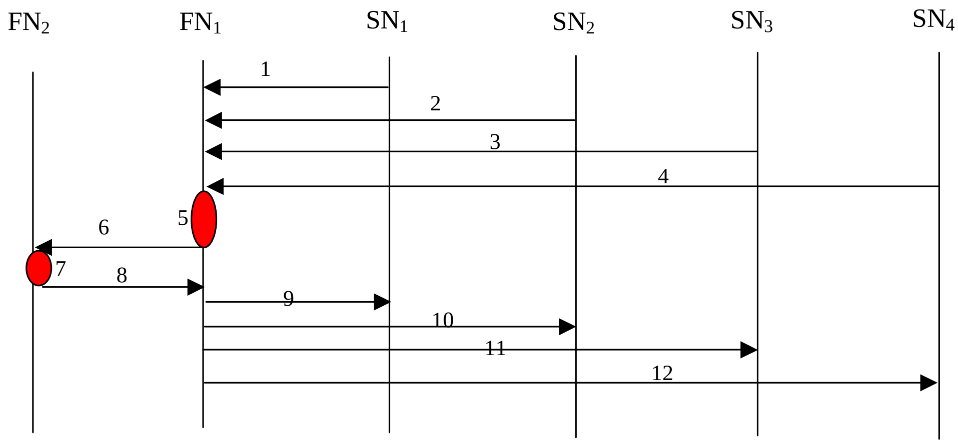

Figure 34 Message flows for gateway/FN implementation

Flow descriptions:

1-4: SN (level 1 Sensor Node) sends one hop link metrics to its FN. The flows do not show necessarily the routing path or message ordering. An efficient forwarding method will be used as determined by routing to reach FN. Anchor nodes report also their coordinates in addition to link metrics.

5: Local maps are computed and merged

6: Send computed maps to the FN that does the global merge. Alternative flow is described in the text below in this subsection.

7: Merge and translate maps. At least three anchor nodes found. Map coordinates are translated to obtain absolute positions.

8: Disseminate location coordinates to corresponding FN.

9-12: Disseminate location information to SNs in the cluster. The flows do not show necessarily the routing path or message ordering. Efficient broadcast can be applied to reach SNs as determined by routing.

An apparent concern for such an implementation would be the overhead on and around FN. FN may be overloaded by messaging and computation requirements, and limited by the battery power. Multi-hop communications are involved in this scheme with traffic concentrating along the paths towards the FN. On the other hand, in SASNet architecture, each SN is already required to send messages to FN and receive from FN during the network formation process, as well as to keep the FN fully aware of its local information. Thus, in step 1-4 as depicted in Figure 34, localization may utilize the network formation messages which are already required by the networking layer.

In step 1-4, each SN sends to FN the following: the node ID's of all the nodes in its local map, and the link distance to each of its one hop neighbour nodes if a ranging technique is used to obtain the measurement. FN, after receiving from all the reported SNs in the cluster computes the local map for each of the nodes in the cluster. FN may need to query the border nodes (or other FN) of its cluster for their two hop neighbour link distance estimates if certain nodes are not in the local FN's cluster. Querying another FN is possible as the node ID shows which FN the SN belongs to. Messaging between FNs may also be more effective using the level-2 communication link of higher bandwidth, compared with messaging between SNs or between the FN and SN. A simplified alternative may employ the FN to compute one map for its entire cluster. This would leave FN a much smaller computation task by using a larger approximated distance matrix. Such a distance matrix includes distances estimated over multiple hops, which may not be accurate. However, if the deployed sensor network is grid like and all the nodes are within a small number of hops apart (e.g., all nodes are within 3 hops from a center node X), the approximated distance matrix may still deliver results meeting the requirements.

After computing all the local maps, FN communicates to other FNs to merge the maps. There may be two options here. In the first option, as with the steps 5 and 6 shown in Figure 34, only a certain FN does the global map merge. All other FNs simply send their local maps or the merged cluster map to this FN. Sending cluster maps incurs a shorter message but leaves fewer alternatives for the merging FN to merge the map. For a merged cluster map, the data includes the node ID and coordinates of each node in the cluster map. Note that this map also has nodes that belong to other clusters but are included in the local map of certain nodes in the local cluster.

Alternatively, each FN will be responsible to merge and translate its own cluster maps. Then the FN would check its cluster map, identify the "foreign" nodes that belong to other clusters and identify the FN X that owns the most "foreign nodes" in its map. The FN X will be contacted and its map fetched to do the merge. Until *3* anchor nodes are found in the FN's current map, the absolute coordinates are obtained for all the nodes in its cluster as well as for some foreign nodes in the merged map. This alternative may result in more map information sent among the FNs at level 2, though it may alleviate the traffic concentration around a certain FN that does the global map merge compared to the first option. At any point, if three anchor nodes are in the merged map, the map is translated to obtain the absolute coordinates. FN then sends to each SN its coordinates or in addition, the local map of a certain sub-neighbourhood to the SN if required. All location computation and map manipulations are performed on FNs.

The advantage of this approach is that it prevents SN from heavy localization computing. Messaging between SN and FN, or between FN and FN may already be required for other purposes thus communication overhead may not be significant. Even without anchor nodes or further map merge among the FN nodes, each FN would have at least a relative map of its own domain. Global, individual and cluster based location information can all be obtained at different levels.

We may compare this option with the one where each SN computes its local map using CCA, illustrated in Figure 35.

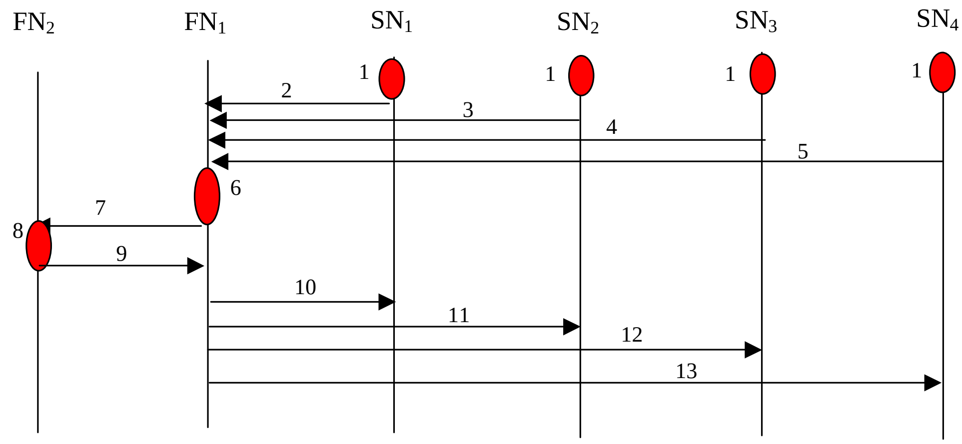

Figure 35 Message flows for gateway/FN implementation with local map computing on SNs

Flow descriptions:

1: Each SN computes its local MAP. An SN needs to send message to certain neighbour X asking the distance between node X and Y where Y is also a neighbour of SN but not necessarily a neighbour of X.

2-5: SN sends local map to its FN. The flows do not show necessarily the routing path or message ordering. Efficient forwarding method will be used as determined by routing to reach FN. Anchor nodes also sends to FN indicating itself having the true coordinates.

6: Local maps are merged on each FN

7: Send maps to the FN that does the global merge. Alternative flow is described in the text below in this section.

8: Merge maps. At least three anchor nodes found. Map coordinates are translated to obtain absolute positions.

9: Disseminate location coordinates to corresponding FN.

10-13: Disseminate location information to SNs in the cluster. The flows do not show necessarily the routing path or message ordering. Efficient broadcast can be applied to reach SNs as determined by routing.

In this option, the messages sent in step 2-5 from SN's to FN include only the node ID's of the nodes in the local map and the coordinates of these nodes. Similar to what was described earlier for the previous approach, the FN does the merge of the local maps (step 6), or alternatively, it may not even merge the local maps. Then the FN can send the merged map (or the received local

maps) to another FN that is supposed to do the global map merge (step 7, 8). After the global map is obtained, the coordinates of each node can be sent to corresponding FN's then to SN's.

This approach distributes the local map coordinates computations over all the SN's that may improve the efficiency of the computing and messaging. The feasibility of doing so needs to be investigated, because an SN may not be able to execute the CCA algorithm due to resource constraints on its CPU, memory and battery power.

In both of the above two options, given *N* sensor nodes in the network, about *2N* or more messages may be needed. The second option requires extra messages between the SNs in the first step, compared with the first option.

We denote the number of required messages for this distributed gateway implementation where the localization executes on the FN as: $N \times p + \theta + N \times p + \delta = 2N \times p + \theta + \delta$.

$N \times p$ - The messages from *N* SN to FN reporting neighbourhood and local distance measurements if any; *p* denotes the average path length between the SN and its FN. Often $p \le 5$ holds. Aggregation of messages from SN to FN may effectively reduce the message volume here. We may piggyback on the routing messages so that these extra messages are not required at all.

θ - The messages between the FNs for map merge. Such messages are transmitted over high capacity links at tier two between FNs.

$N \times p$ - The messages sent from the FN nodes conveying the location coordinates. Assume one separated message is sent to each of the *N* SN here, though multicast can effectively reduce this message volume. We may piggyback on the routing messages so that these extra messages are note required at all.

The δ element includes the messages between SNs when the local maps are computed on the SNs as in the second option described above.

4.1.2 Fully Distributed Cooperative Localization

A yet even more distributed implementation of CCA-MAP, the fully distributed approach can be devised [45]. In this approach, every node including SN and FN computes their own local map. Then a selected starting node, e.g., an anchor node queries all its neighbour nodes of the node IDs included in their own local maps. The starting node, which we name here the current node *CN*, would select the node N_x such that the local map of N_x and the local map of *CN* share the most number of common nodes. *CN* sends its local map to node N_x . Node N_x merges the received map with its own map and then becomes the current node *CN*. The new *CN* then selects a new node N_x from its "neighbour" nodes as described before by checking common nodes in its current map and other nodes' local maps, excluding the nodes that have already acted as *CN*. This merging process propagates in the network until the map is merged at the last node in the network. This approach requires each node to compute its local map and merge its local map into the global map. To merge the map for a network of *N* nodes, *N-1* messages will be sent with the data size of a growing map. The data size is *N* coordinates *(2N* or *3N)* maximum. Message size can be of concern in this option as it grows. A large number of messages may be required for the node *CN*

to select the node N_x. These messages are sent acquiring the node ID's included in other node's local map so that N_x can be identified. With the growth of the current map, messages across multiple hops will be needed as the node that is sought after may not be one hop away from the *CN*. Different options can be considered here to minimize the messaging. One option is to have a few nodes in the network that have the list of node IDs in all the local maps of all the nodes. Then the *CN* only needs to query one of such archived nodes to find the next N_x. Other options [45] proposed include to always select N_x among the neighbours of CN or if none is found, to simply forward the current map to one of the neighbours that is most likely to find a N_x.

Local messages in addition are required for each node to reach the two hop neighbour nodes, acquiring their link metrics and their distance approximation to other nodes. This may partially reuse the local routing keep-alive messages, with some additional parameters to query distances if needed.

The number of messages required in the fully distributed implementation may be denoted as $\delta + (N-1)\times\varphi + (N-1)\times p'$.

δ - The sages exchanged for computing local maps on the SNs, often far fewer than the number of nodes covered in the local map as most information already known from routing messages.

$(N-1)\times\varphi$ - The message needed to discovery which node can be selected as the next node in the map merge. Each node needs in average φ message to find the next merging node.

$(N-1)\times p'$ - The messages used to pass the currently merged map from the first node all the way to the last node to complete the map merge, where *p'* is often "1" or quite small.

4.1.3 Iterative Localization

Since iterative localization often executes in a fully distributed manner, it would be interesting to compare the fully distributed cooperative algorithm with the iterative implementation. For iterative algorithms, the general messaging process of the iterative localization algorithm is presented in Figure 36 below for comparison.

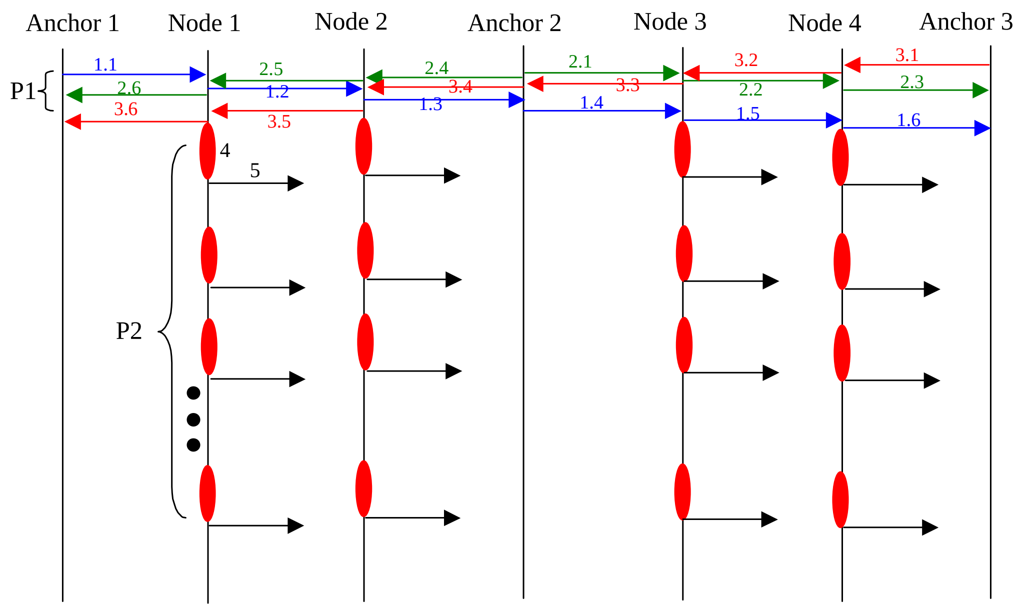

Figure 36 Messaging in iterative localization

Flow Descriptions:

Phase 1(P1) – Initialization from anchor nodes

1.1-1.6: Anchor node 1 broadcasts its coordinates. The flows do not illustrate necessarily the message route. Efficient broadcast or limited flooding will be used to control the flows.

2.1-2.6: Anchor node 2 broadcasts its coordinates. The flows do not illustrate necessarily the message route. Efficient broadcast or limited flooding will be used to control the flows.

3.1-3.6: Anchor node 3 broadcasts its coordinates. The flows do not illustrate necessarily the message route. Efficient broadcast or limited flooding will be used to control the flows.

There is no ordering imposed on the above messages. That is, anchor 1 is not necessarily sending its message first.

Phase 2(P2) – Iterative Localization

4: Each node upon gathering distance information of up to at least three nodes with known coordinates computes its location and estimates the position error. Always tries to find the best possible result using the known locations with least predicated error in computing.

5: Each node sends to its one-hop neighbours its calculated position and estimated error if the calculated position is estimated more accurately than last time.

Flow 4 and 5 are carried on each node repeatedly until position calculations stabilize. This may require 6 or more rounds.

If comparing Figure 36 with Figure 34 and Figure 35 of cooperative localization schemes, iterative localization may incur local messages with smaller message size (coordinates of a single node). The message needed may be denoted as $K \times N + N \times l$.

$K \times N$ - K anchor nodes broadcast location coordinates to all the N nodes in the network using flooding. If using efficient broadcast instead of flooding, this message amount can be reduced.

$N \times l$ - Each sensor node send to its neighbour nodes about its currently computed coordinates in l iterations.

In cooperative localization, fewer messages may be sent with concentrated direction and larger message size (local map and merged map both encompass information of multiple nodes). The following Table 1 summarizes the above preliminary analysis on the message requirements.

Table 1: Estimated Message Requirements

Approaches	**Distributed FN gateway cooperative**	**Fully distributed cooperative**	**Iterative (fully distributed)**
Messages	$2N \times p + \theta + \delta$	$\delta + (N-1) \times \varphi$ $+ (N-1) \times p'$	$K \times N + N \times l$
Notions	N : total number of SN p: average path from the SN to its FN θ: message exchanged between FNs over tier 2 high capacity links δ : message required for computing local map, often far smaller than the local map size	N : total number of SN p': average path length when passing merged maps δ : message required for computing local map, often far smaller than the local map size φ: average messages needed by each node to find the next node for map merge.	K : number of anchor nodes N : total number of nodes l : number of iterations required
Comments	$\delta \geq 0$; $2N \times p$ may be eliminated by piggybacking on routing messages	$\delta > 0$; p' often “1” or close to “1”	$l \geq 6$ often required; $K \times N$ can be reduced by using efficient broadcast (e.g., MPR forwarding)

At this stage, we have not yet decided on the details of the platforms of SN and FN. Though impossible to judge the capability of the SN node at a detailed level, an analysis of the CCA implementation complexity will assist us later in making the decisions. In the next subsection, we analyze the implementation complexity of the CCA algorithm.

4.2 CCA Implementation Analysis

4.2.1 Memory Usage

A preliminary analysis of major memory usage for implementing CCA localization scheme is presented here. The analysis here is based only on our current understanding of the options and will be updated while we refine the SASNet architecture and design options.

The memory usage for the computation of each local map using CCA:

- Distance matrix of local map. Given the local map size of *K*, the distance matrix is $D = K \times K$. As *K(i,j)=K(j,i)*, the required memory need to at least hold $K^2/2$ entries. (Note: these estimates are preliminary and will be updated when more information is gathered.) During our experiments, for a grid network with node connectivity level around 14, we have observed $K_{\max} < 40$. Normally, network should have a smaller node connectivity level which would result in smaller *K*. Random networks on the other hand can have a much higher $K_{\max}$ at the same connectivity level.
- Coordinates of nodes in the local map. Given *K* nodes in the local map, it requires memory to hold $K \times d$ items, where *d* is the dimension of *2* (*2D* space) or *3* (*3D* space). Because each cycle manipulates the results from its previous cycle, two of such arrays may be needed.
- In each cycle, *K-1* new inter-node distance values need to be computed and held. This memory is reused in each computing cycle. We also take *K-1* distance values from the distance matrix *D* for comparison. It may require memory for these *K-1* entries taken out of *D*, though it may not be at all necessary depending on the implementation.

4.2.2 Computing Complexity

During the CCA computation, the following operations, mostly mathematical are executed in each cycle.

- Compute *K-1* distance values between *K* nodes. This involves computing of square root.
- Computing of the weighing function. We used exponential function in Matlab experiments. Simpler functions of the bubble function for example may be used instead. An option is to pre-compute the function and pre-assign the values that

should be used during looping of cycles. The disadvantage of such an option is the memory consumption to store the pre-computed values.

4.3 MDS Implementation Analysis

In this subsection, we analyze the implementation requirements for MDS-MAP for comparison. To implement the MDS localization scheme, a local map for each node in the network needs to be computed using the MDS data reduction algorithm.

4.3.1 Memory Usage

MDS performs singular value decomposition on the distance matrix. The memory usage for the computation of each local map using MDS may include the following:

- Distance matrix of local map. Given the local map size of *K*, the distance matrix is $D = K \times K$. As *K(i,j)=K(j,i)*, the required memory need to at least hold K^2*/2* entries. In MDS algorithm, each node covers its two hop neighbours in its local map. During our experiments, for a grid network with node connectivity level around 14, we have observed $K_{\max} < 65$. Normally, network should have a smaller node connectivity level which would result in smaller *K*.
- Because MDS-MAP takes two-hop neighbourhood to form local map, so the average size of the local map is larger than that of CCA-MAP. As an example, for the irregular network of *64* nodes used in our experiment, at *CL=14*, the MDS local map has an average size of *39.3* and the CCA local map *20.3*.
- Coordinates of nodes in the local map. Given *K* nodes in the local map, it requires memory to hold $K \times d$ items, where *d* is the dimension of *2* (*2D* space) or *3* (*3D* space).
- To perform singular value decomposition, eigenvalues and eigenvectors need to be kept. This would require matrix of $K \times 3$ and 3×3 as only the first three items are selected. Sorting of the eigenvalues may require a few more arrays of small sizes which won't be significant.

At this stage, we have not searched for the best way to compute the eigenvalues and eigenvectors for the distance matrix of the local map on either SN or FN, until we make decisions on the platforms of these nodes. Therefore, the memory size required for the computing steps of eigenvalues and eigenvectors is not estimated here.

4.3.2 Computing Complexity

MDS location mapping performs singular value decomposition on the approximated distance matrix *D*. For the squared distance matrix, this is to compute the eigenvalues and eigenvectors of *D*. Often one can compute all the eigenvalues of matrix *D* by solving the equation $det(D - \lambda I) = 0$, where *I* is the unit matrix. This system has a degree of *K* and *K* eigenvalues. In the algorithm, we only need the first 3 for position estimation. The solution to such as system may resort to the

least square optimization problem. The practical implementation for solving eigenvalues depends on the selected node platform, its computing capabilities and the math function libraries it offers. MDS requires a suitable solution for computing eigenvalues and eigenvectors on the selected platform.

The Table 2 below summarizes the above preliminary analysis on computing requirements.

Table 2 Estimated Computing Requirements

Approaches	**Local Map Computing in CCA-MAP**	**Local Map Computing in MDS-MAP**
Memory	$\frac{K^2}{2} + 2K \times d + 2(K-1)$	$\frac{K^2}{2} + K \times d + d \times d + \delta$
Notions	*K*: local map size *d*: Dimension required (*d=2* or *d=3*)	*K*: local map size *d*: Dimension required (*d=2* or *d=3*) δ : extra memory required when computing singular value decomposition
Computing	Square root; weighted function (e.g., exponential function)	SVD computing
Complexity	$O(K^2)$	$O(K^3)$

4.4 Map Merging

Either CCA-MAP or MDS-MAP would need to merge local maps together into the global map. If the local maps are merged on a selected gateway node such as FN, *m* local maps need to be stored on FN during the merge process, where *m* is the number of nodes in the cluster (subnetwork). Assuming *k* nodes contained in each local map, coordinates of *mk* nodes (e.g., 2*mk*) need to be stored for local maps. A copy of the resulting current map is needed where the size of the current map grows into *m* during the merging process. If the map merging is done on each node in a fully distributed manner, a copy of the current map needs to be stored in addition to the local map of the local node during the process. The current map grows into size *N* where *N* is the number of all the nodes in the network (i.e., global map). CCA-MAP and MDS-MAP apply the same map merging process. Thus there is little difference in terms of implementation here.

5 Conclusions

In this report, we have presented our study of the localization solution options for SASNet, a Self-healing Autonomous Sensor Network for support of military operations. Particularly, the cooperative localization schemes were examined as recommended from our previous studies on WSN localization for SASNet. The cooperative localization scheme delivers highly accurate position estimate results using “zero” or a minimum number of anchor nodes.

We have found a class of non-linear reduction methods using efficient neural network techniques, namely the Curvilinear Component Analysis (CCA) that can map node positions based on the distance matrix with a high level of accuracy and relatively low computational cost. Applying the CCA data reduction method, we devised the CCA-MAP algorithm, a distributed cooperative localization scheme generating location maps for WSN.

Taking SASNet deployment scenarios, we compared the performance measurements of CCA-MAP and MDS-MAP in calculating node positions for these scenarios, using both range-based and range free options. In all the network configurations simulated where ranging capability is employed, at relatively low network connectivity levels, the CCA-MAP algorithm can achieve accurate node position estimates with an average error of less than *10%r*, using a minimum number of anchor nodes. Using connectivity information only, CCA-MAP can also contain the position estimation error within *20%r* in all the network configurations tested. We have found that CCA-MAP delivers better position accuracy compared with MDS-MAP. We have also studied the effect of anchor node positions. At relatively low connectivity levels, anchor node positions affect the position estimation results. Merely increasing a small number of anchor nodes may not improve the accuracy. At high connectivity levels, the differences between different anchor node positions start to diminish. Increasing the number of anchor nodes also decrease the differences generated by different anchor positions.

Through the scenario studies, we have found that random networks of irregular shape such as a “narrow C” with scattered nodes poses the challenge for both CCA-MAP and MDS-MAP. In fact, both CCA-MAP and MDS-MAP achieve much better results compared with other schemes reported for such anisotropic networks. The only known solution to this type of network would require a large number of anchor nodes in the network.

The drawback of cooperative algorithms investigated in this study would be their computational complexity, if implemented on each of the sensor node. This is a possible pitfall for the CCA-MAP scheme, especially in the range-free scenarios when the distance information is not available but approximated by the network connectivity information. We have elaborated the different implementation options for CCA-MAP, conducted the analysis of its implementation requirements and compared it with other related approaches such as the MDS-MAP and the iterative localization scheme.

The computational cost of CCA-MAP is not fully determined yet until it is implemented in C or another real platform programming language. If the computational cost of CCA-MAP can be handled by the selected platform, CCA-MAP is a quite viable solution that delivers accurate position estimates using a minimum number of anchor nodes.

References

[1] SASNet Requirement Document.

[2] SASNet Networking Architecture Document

[3] SASNet Demonstration Plan

[4] Li Li, Localization in Self-healing Autonomous Sensor Network (SASNet): Requirements, Design Considerations and Research Recommendations, Report of CRC/DRDC , CRC-RP-2007-005, September 2007

[5] Li, L and Kunz, T, "Localization Applying An Efficient Neural Network Mapping", *in proceedings of First International Conference on Autonomic Computing and Communication Systems*, Rome, Italy, October, 2007

[6] Li, L and Kunz, T, "Cooperative Node Localization for Tactical Wireless Sensor Networks", *in proceedings of IEEE MILCOM'07*, Orlando, FL, October, 2007

[7] Yasser Gadallah, Data Dissemination in Wireless Sensor Networks: Requirements, Design Considerations and Research Recommendations, Technical Memorandum DRDC Ottawa TM 2006-134, March 2006

[8] K.Langendoen and N.Reijers, "Distributed localization in wireless sensor networks: a quantitative comparison." *Elsevier Computer Networks*, 43(4): 499-518, 2003

[9] Lance Doherty, Kristofer Pister and Laurent El Ghaoui. "Convex Position Estimation in Wireless Sensor Networks", *in proceedings of IEEE INFOCOM 2001*, Anchorage, Alaska, April 2001.

[10] Yi Shang, Wheeler Ruml, Ying Zhang, Markus Fromherz. "Localization from Mere Connectivity", *in Proceedings of MobiHoc 2003*, Annapolis, MD, June 2003

[11] Yi Shang and Wheeler Ruml, "Improved MDS Based Localization", *in Proceedings of IEEE INFOCOM 2004*, Hong Kong, March, 2004

[12] B.H.Wellenhoff, H.Lichtenegger and J. Collins, *Global Positioning System: Theory and Practice*, Fourth Edition, Springer Verlag, 1997

[13] Andreas Savvides, Chih-Chieh Han, Mani B. Strivastava. "Dynamic fine-grained localization in Ad-Hoc networks of sensors", *in Proceedings of MobiCom 2001*, Rome, Italy, July 2001

[14] N. B. Priyantha, A. Chakraborty and H. Balakrishnan, "The Cricket Location-Support System", *in Proceedings of MobiCom'00*, New York, August 2000.

[15] D. Niculescu and B. Nath, “Ad Hoc Positioning System (APS) using AoA”, *in Proceedings of INFOCOM’03*, San Francisco, CA, 2003

[16] P. Bahl and V. N. Padmanabhan, “RADAR: An In-Building RF-Based User Location and Tracking System”, *in Proceedings of the IEEE INFOCOM’00*, Tel-Aviv, Isreal, March 2000.

[17] J. Hightower, G. Boriello and R. Want, *SpotON: An indoor 3D Location Sensing Technology Based on RF Signal Strength*, University of Washington CSE Report #2000-02-02, February 2000.

[18] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin and S. Wicker, *Complex Behavior at Scale: An Experimental Study of Low-Power Wireless Sensor Networks*, Technical Report UCLA/CSD-TR 02-0013, 2002.

[19] KLEINROCK, L. AND SILVESTER, J., “Optimum transmission radii for packet radio networks or why six is a magic number”, in *Proceedings of the IEEE National Telecommunications Conference*, Birmingham, Alabama, December 1978.

[20] L. Girod and D. Estrin, “Robust Range Estimation using Acoustic and Multimodal Sensing”, *in Proceedings of IROS’01*, Maui, Hawaii, October 2001.

[21] C. Savarese, J. Rabay and K. Langendoen, “Robust Positioning Algorithms for Distributed Ad-Hoc Wireless Sensor Networks,” *in Proceedings of USENIX Technical Annual Conference*, Monterey, CA, June 2002.

[22] A. Savvides, H. Park and M. Srivastava, “The Bits and Flops of the N-Hop Multilateration Primitive for Node Localization Problems”, *in Proceedings of First ACM International Workshop on Wireless Sensor Networks and Application*, Atlanta, GA, September 2002.

[23] K. Whitehouse and D. Culler, “Calibration as Parameter Estimation in Sensor Networks”, *in Proceedings First ACM International Workshop on Wireless Sensor Networks and Application*, Atlanta GA, September 2002.

[24] Dragos Niculescu and Badri Nath, “Ad Hoc Positioning System (APS)”, *in Proceedings GLOBECOM 2001*, San Antonio, Texas, November 2001

[25] Nirupama Bulusu, John Heidemann and Deborah Estrin. “GPS-less Low Cost Outdoor Localization for Very Small Devices”, *IEEE Personal Communications Magazine*, Vol.7, No.5, October, 2000, pp.28-34.

[26] Nirupama Bulusu, John Heidemann and Deborah Estrin. “Density Adaptive Algorithms for Beacon Placement in Wireless Sensor Networks”, *in Proceedings of IEEE ICDCS 2001*, Phoenix, Arizona, April 2001.

[27] Tian He, Chengdu Huang, Brian M. Blum, John A. Stankovic, Tarek Abdelzaher. “Range-free Localization Schemes for Large Scale Sensor Networks”, *in Proceedings of MobiCom 2003*, San Diego, Ca. September, 2003

[28] Dragos Niculescu and Badri Nath. "DV Based Positioning in Ad hoc Networks", *Kluwer Journal of Telecommunication Systems*, Vol 22:1-4, pp.267-280, 2003.

[29] Radhika Nagpal, Howard Shrobe, and Jonathan Bachrach. "Organizing a Global Coordinate System from Local Information on an Ad Hoc Sensor Network", *in Proceedings of 2nd International Workshop on Information Processing in Sensor Networks (IPSN)*, Palo Alto, CA., April 2003.

[30] Thomas Kleine-Ostmann and Amy E. Bell, "A Data Fusion Architecture for Enhanced position Estimation in Wireless Networks", *IEEE Communication Letters*, Vol 5, No. 8, August, 2001, pp.343-345

[31] Lingxuan Hu et al, "Localization for Mobile Sensor Networks", *in Proceedings of MobiCom'04*, Philadelphia, PA, September 2004

[32] Jeffrey Hightower and Gaerano Borriello, "Location Systems for Ubiquitous Computing", *IEEE Computer*, Vol.34, No.8, Aug. 2001, pp.57-66,

[33] N. Bulusu, J. Heidemann, D. Estrin, and T. Tran, "Self-Configuring Localization Systems: Design and Experiemental Evaluation", *ACM Trans. On Embedded Computer Systems*, May 2003, Vol. 3, No.1, Feb. 2004, pp.24-60

[34] N. Patwari, J.N.Ash, S. Kyperountas, A.O.Herro, R.L.Moses, and N.S. Correal, "Locating the nodes", *IEEE Signal Processing Magazine*, Vol. 22, No.4, July 2005, pp.54-69

[35] S. Gezici, Z. Tian, G. Giannakis, H. Kobayashi, A. Molisch, H. Poor and Z. Sahinoglu, "Localization via Ultra-Wideband Radios", *IEEE Signal Processing Magazine*, Vol. 22, No.4, July 2005, pp.70-84

[36] G. Sun, J. Chen, W. Guo and K. Liu, "Signal Processing Techniques in Network-Aided Positioning", *IEEE Signal Processing Magazine*, Vol. 22, No.4, July 2005, pp.12-23

[37] R.J. Fontana, E. Richley, and J. barney, "Commercialization of an Ultra Wideband Precision Asset Location System", *in Proceedings of IEEE Conference on Ultra Wideband Systems and Technologies*, Reston, VA, Nov. 2003

[38] J. Albowicz, A. Chen and L. Zhang, "Recursive Position Estimation in Sensor Networks", *in proceedings of IEEE International Conference on Network Protocols*, Nov. 2001, pp.35-41.

[39] S. Guha, R. Murty and E. Sirer, "Sextant: A Unified Node and Event Localization Framework using Non-Convex Constraints", *in Proceedings of MobiHoc'05*, Urbana-Champaign, IL, May 2005.

[40] N.B.Priyantha, H. balakrishnan, and S. Teller, "Anchor-free distributed localization in sensor networks", *in Proc. 1st International Conference on Embedded Networked Sensor Systems (SenSys 2003)*, Los Angeles, CA, Nov. 5-7, 2003, pp340-341

[41] A. Youssef, A. Agrawala and M. Younis, “Accurate Anchor-free Node Localization in Wireless Sensor Networks”, *in Proceedings of IEEE Workshop on Information Assurance in Wireless Sensor Networks (WSNIA),* April, 2005, Pheonix, AZ.

[42] H.V. Poor, *An Introduction to Signal Detection and Estimation*, 2nd ed. NewYork, Springer-Verlag, 1994.

[43] N. Patwari, et., “Relative Location Estimation in Wireless Sensor Networks”, *IEEE Trans. Signal Processing*, Vol. 51, No. 8, Aug. 2003, pp.2137-2148,

[44] Pierre Demartines and Jeanny Herault, Curvilinear Component Analysis: A Self-Organizing Neural Network for Nonlinear Mapping of Data Sets”, *IEEE Trans. on Neural Networks*, Vol. 8, No.1, Jan. 1997, pp.148-154

[45] Yi Shang, Wheeler Ruml, Ying Zhang, Markus Fromherz, “Localization from Connectivity in Sensor Networks”, *IEEE Trans. on Parallel and Distributed Systems*, Vol.15, No.11, Nov. 2004, pp.961-974

[46] Ji, X. and Zha, H. “Sensor positioning in wireless ad-hoc sensor networks using multidimensional scalin”, *in Proceedings of IEEE INFOCOM 2004*, Hong Kong, China, March, 2004

[47] Pathirana, P., Bulusu, N., Savkin, A. and Jha, S. “Node localization using mobile robots in delay-tolerant sensor networks”, *IEEE Trans. on Mobile Computing*, Vol. 4, No. 3, May/June 2005, pp.285-296

[48] P. Drineas, A. Javed, M. Magdon-Ismail, G. Pandurangan, R. Virrankoski and A. Savvides, “Distance Matrix Reconstruction from Incomplete Distance Information for Sensor Network Localization”, *in Proceedings of the Third Annual IEEE Communications Society Conference on Sensor, Mesh, and Ad Hoc Communications and Networks (SECON'06)* , Sept., 2006, Reston, VA, USA.

[49] V. Vivekanandan and V.W.S. Wong, “Ordinal MDS-based Localization for Wireless Sensor Networks”, *International Journal of Sensor Networks*, Vol.1, No. 3/4, 2006, pp.169-178

[50] Y. Zhang, Q. Huang and J. Liu, “Sequential Location Algorithm for Active Sensor Network Deployment", *in Proceedings of IEEE PCAC 2006*, April 2006, Vienna, Austria

[51] I. Borg and P. Groenen, *Modern Multidimensional Scaling, Theory and Applications,* New York: Springer-Verlag, 1997

[52] W. S. Torgeson, “Multidimensional scaling of similarity,” *Psychometrika*, Vol. 30, pp. 379–393, 1965

[53] J.W. Sammon, "A nonlinear mapping for data structure analysis", *IEEE Trans. Comput*., Vol. C-18, Issue 5, 1969, pp.401-409

[54] J. Liu, Y. Zhang and E. Zhao, "Robust Distributed Node Localization with Error Management", *in Proceedings of MobiHoc'06*, May 2006, Florence, Italy.

[55] Ssu, K-F., Ou, C-H.and Jiau, H. (2005) "Localization with mobile anchor points in wireless sensor networks", *IEEE Transactions on Vehicular Technology*, Vol. 54, No. 3, May, 2005, pp.1187-1197

[56] Chintalapudi, K.K. Dhariwal, A. Govindan, R. Sukhatme, G., "Ad-hoc Localization using Ranging and Sectoring", *in Proceedings of IEEE INFOCOM 2004*, Hong Kong, March, 2004

[57] T. Eren, D.K. Goldenberg, W.Whiteley, Y.R. yang, A.S. Morse, B. D. O. Anderson and P.N. belhumeur, "Rigidity, Computation, and Randomization in Network Localization", *in Proceedings of IEEE INFOCOM 2004*, Hong Kong, March, 2004

[58] P. Biswas and Y. Ye, "Semi-definite programming for ad hoc wireless sensor network localization," *in Proceedings of IPSN 2004*, Berkeley, CA, April 2004

[59] Sitao Wu and Tommy W. S. Chow, "PRSOM: A New Visualization Method by Hybridizing Multidimensional Scaling and Self-Organizing Map", *IEEE Trans. On Neural Networks*, Vol.16, No.6, Nov. 2005, pp.1362-1380

[60] H. Yin, "ViSOM – A Novel Method for Multivariate Data Projection and Structure Visualization", *IEEE Trans. On Neural Networks*, Vol.13, No.1, Jan. 2002, pp.237-243

[61] *Crossbow MICA2 sensor node data sheet* http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICA2_Datasheet.pdf

List of symbols/abbreviations/acronyms/initialisms

DRDC	Defence R&D Canada
CRC	Communications Research Centre
2-D	Two Dimensional
3-D	Three Dimensional
AFL	Anchor-Free Localization
AHLoS	Ad Hoc Localization System
AOA	Angle Of Arrival
APIT	Approximated Point-In-Triangulation Test
APS	Ad-Hoc Positioning System
CRB	Cramér-Rao Bound
DV	Distance Vector
DR	Dead-Reckoning
GPS	Global Positioning System
IC	Integrated Circuit
ID	Identifier
IEEE	Institute of Electrical and Electronics Engineers
IR	InfraRed
LOS	Line-Of-Sight
LS	Least Square
MC	Monte Carlo
MDS	Multi-dimensional Scaling
MEMS	Micro-Electro-Mechanical Systems

ML	Maximum Likelihood
MLE	Maximum Likelihood Estimation
MMSE	Min-Mean Square Error estimation
MP	Multi-Path
NLOS	Non-Line-Of-Sight
NP	Non-deterministic Polynomial time
RF	Radio Frequency
RSSI	Radio Signal Strength Indicator
SASNet	Self-healing Autonomous Sensing Network
SIS	Sequential Importance Sampling
SNR	Signal-to-Noise Ratio
TDOA	Time Difference Of Arrival
TOA	Time Of Arrival
UWB	Ultra-Wide Band
WSN	Wireless Sensor Network

Distribution list

Document No.: DRDC Ottawa TR 2008-020

	LIST PART 1: Internal Distribution by Centre
2	DRDC Ottawa Library
2	TOTAL LIST PART 1
	LIST PART 2: External Distribution by DRDKIM
3	Library and Archives Canada (2 HC, 1CD)
1	Joe Schlesak, Communication Research Centre
1	Louise Lamont, Communication Research Centre
1	CRC library
1	Ying Ge, Communication Research Centre
1	Li Li, Communication Research Centre
1	Luc Boucher Communications Research Centre
1	Thomas Kunz, Carleton University, 1125 Colonel By Drive25 Colonel, Ottawa, ON, Canada K1S 5B6
1	Benoit Ricard DRDC ValCartier
1	David Waller Constitution Building, 305 Rideau Street, Ottawa, ON, Canada K1A 0K2
1	Craig Maskell, Constitution Building, 305 Rideau Street, Ottawa, ON, Canada K1A 0K2
1	Shawn Hoag, Constitution Building, 305 Rideau Street, Ottawa, ON, Canada K1A 0K2
1	DRDKIM (CD)
15	TOTAL LIST PART 2
17	**TOTAL COPIES REQUIRED**

This page intentionally left blank.

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1. ORIGINATOR (The name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.) Defence R&D Canada – Ottawa 3701 Carling Avenue Ottawa, Ontario K1A 0Z4	2. SECURITY CLASSIFICATION (Overall security classification of the document including special warning terms if applicable.) UNCLASSIFIED

3. TITLE (The complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title.)

Localization in self-healing autonomous sensor networks (SASNet): Studies on cooperative localization of sensor nodes using distributed maps

4. AUTHORS (last name, followed by initials – ranks, titles, etc. not to be used)

Li, L

5. DATE OF PUBLICATION (Month and year of publication of document.) January 2008	6a. NO. OF PAGES (Total containing information, including Annexes, Appendices, etc.) 80	6b. NO. OF REFS (Total cited in document.) 61

7. DESCRIPTIVE NOTES (The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)

Technical Report

8. SPONSORING ACTIVITY (The name of the department project office or laboratory sponsoring the research and development – include address.)

Defence R&D Canada – Ottawa
3701 Carling Avenue
Ottawa, Ontario K1A 0Z4

9a. PROJECT OR GRANT NO. (If appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant.) 12pk	9b. CONTRACT NO. (If appropriate, the applicable number under which the document was written.)
10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document.) DRDC Ottawa TR 2008-020	10b. OTHER DOCUMENT NO(s). (Any other numbers which may be assigned this document either by the originator or by the sponsor.)

11. DOCUMENT AVAILABILITY (Any limitations on further dissemination of the document, other than those imposed by security classification.)

Unlimited

12. DOCUMENT ANNOUNCEMENT (Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11) is possible, a wider announcement audience may be selected.))

Unlimited

13. ABSTRACT (A brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual.)

The Self-healing Autonomous Sensing Network (SASNet) presents an advanced Wireless Sensor Network (WSN) that aims to enhance the effectiveness of mission operation in the contemporary military environment, by providing relevant and accurate situational awareness information. In order to achieve this objective, precise location information is required in SASNet. In this report, we present the studies conducted in the SASNet project on cooperative localization algorithms for wireless sensor nodes. We have taken the cooperative localization approach following the recommendations of the survey study conducted last year, which identified that cooperative localization schemes can often produce accurate results using a very small number of anchor nodes or even no anchor nodes.

The cooperative localization scheme adopted in this study computes a local map for each sensor node using all the available link metric constraints, and then merges the local maps into a global map where each node acquires its location coordinates. In the study, we examined the advanced techniques of non-linear data mapping for computing the local maps from the large data set of link constraints. In particular, we selected a non-linear mapping technique, the Curvilinear Component Analysis (CCA) from a class of highly efficient neural networks and applied it to WSN localization, proposing a novel cooperative localization scheme based on CCA. We studied CCA localization in comparison with another leading cooperative localization scheme, namely the MDS (Multi-Dimensional Scaling) map method.

In the report, we first briefly review the related work on WSN localization and re-examine the pros and cons of the selected cooperative approach vs. other approaches, most notably the iterative approach using trilateration. We then describe the CCA algorithm for data non-linear mapping, and extend it to solve the problem of sensor node position estimation. A detailed elaboration of the proposed CCA-MAP localization scheme is given. The performance simulations of CCA-MAP are conducted using SASNet scenarios and their results are illustrated and compared with the MDS-MAP algorithm, which is a leading cooperative localization scheme published in the literature. From the simulation experiments, advantages and shortcomings of the CCA-MAP algorithm are analyzed. Further, we discuss the design considerations of the discussed cooperative localization algorithms to compare and examine their implementation feasibility. Finally, conclusions and recommendations from this study are presented.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (Technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

wireless sensor networks, node localization, non-linear data mapping, curvilinear component analysis, multi-dimensional scaling, distributed computing

Defence R&D Canada

Canada's leader in Defence
and National Security
Science and Technology

R & D pour la défense Canada

Chef de file au Canada en matière
de science et de technologie pour
la défense et la sécurité nationale

DEFENCE DÉFENSE

www.drdc-rddc.gc.ca